SONY®

MSX
Technical
Reference Document

by L. Theunissen

SONY®

MSX

Technical

Reference Document

by L. Theunissen

INTRODUCTION

This booklet contains a technical explanation of 8-bit home computer hard- and software based on the MSX-standard, and examples of practical circuitry, as used on the Sony Hit-Bit series microcomputers HB-55 and HB-75. It is intended as an introduction to the MSX-format for engineers and technicians with a solid basic knowledge of microcomputer circuitry in general, and the Z-80 CPV in particular. It will help them understand completely the philosophy behind the MSX-format and the technical solutions used to come to a common, fully interchangeable format that will be used by many computer manufacturers worldwide.

December 1984
Sony Service Centre (Europe) N.V.

CONTENTS

INTRODUCTION TO MSX

The low end of the home-computer market has now a worldwide standard, called MSX.
MSX, developed by the Microsoft Corporation, is both a hardware and a software standard for 8-bit personal computers.

This implies that all the MSX computer models, marketed by more than a dozen manufacturers, will share the software compatibility attribute.

The hardware in this MSX standard is implemented with three popular chips:

- **the Z80 Central Processing Unit** (Zilog)
- **the 9918 Video Display Processor** (Texas Instruments)
- **the 8910 Sound Generator** (General Instruments)

Included are an Atari joystick interface and specifications for a, read only memory based, cartridge.

The built-in software is written in a new version of Microsoft Basic, called Super-Extended Microsoft Basic. This Basic language has more of a monitor than of an operating system, in that it cannot control disk drives.
But MSX has enough command structure to control the cartridges and cassette-tape recorder that home computers use.

A possible result of this new standard may be that makers of home appliances and TV sets may be prompted to build in standard input/output ports and processors to connect to MSX units, simplifying the introduction of control systems into the home.

Although MSX is originally created as a standard to run ROM-based software, it is now ready in a Disk-Operating-System version.
Designed for use with 8-bit Z80 microprocessor hosts, MSX-DOS emulates CP/M-80 calls and contains MS-DOS file formats.
In this respect, it is kind of the best of both worlds (8-bit CP/M from Digital Research and 16-bit MS-DOS from Microsoft).

Using Microsoft's words to conclude, MSX means a "Home Personal Computer System for Everybody".

HARDWARE SPECIFICATIONS

1. Summary of the Hardware Requirements

1-1. Specifications

1-1-1. Required Components

- CPU   4MHz Z-80A compatible

- Memory   ROM 32K (MSX system software)
                     RAM minimum 8K (16K* recommended)

- Screen Display   Text display capability 40 x 24 (refer to section 2.4)
                     Graphic 256 x 192
                     Colour 16

- Cassette tape   FSK format 1200/2400 Baud

- Sound   8 Octave, 3 Voices

- Character Set   Alphanumeric, Japanese, Graphic (Japanese Version)
                     Alphanumeric, European, Graphic (International Version).

- Keyboard   U.S./Europe, French*, German*, Japanese

- Expansion Slot   Software cartridge, expansion BUS slots

- Joystick   1 or 2

1-1-2. Recommended Extensions for U.S./Europe

- Memory   RAM 64K* total

- Expansion Slot   Second

- Video   RF output

* Items with asterisk may not be built-in in the minimum system.

1-1-3. Standardized Optional Extensions*

- Screen Display*   80-Column text

- Clock*   Battery backed-up CMOS

- Communication*   RS-232

- Floppy Disk*   According to each company. Format is MS-DOS compat-
                    ible

- Printer*   8 bit parallel

2. Components

2-1. LSI

- CPU   Z-80 Compatible
                    CLOCK 3.579545MHz (NTSC Colour Sub Carrier Frequency)
                    1 WAIT in M1 CYCLE

- VDP   TI TMS-9918A Compatible

- PSG   GI AY-3-8910 Compatible

- PPI   Intel i-8255 Compatible

2-2. Memory

- ROM   MSX BASIC 32KB

- RAM   8KB or more

                    Basic unit has four logical slots, so the total
                    memory space can be expanded up to 256KB. Each
                    logical slot can be expanded to have up to 4 physical
                    slots, or 16 slots. In this case, the total memory
                    space is 1 megabyte.

                    BASIC ROM occupies address 0 to 7FFF, RAM address
                    starts from FFF and grows downward on the memory map.

                    Although the BASIC ROM only uses 32K of RAM, the
                    machine needs 64K of RAM for MSX-DOS.

* Items with asterisk may not be built-in in the minimum system.

2-3. Interrupt

- NMI   Not used. See note.

- INT   Accept interrupts from VDP and cartridge. The
                  interrupt is Z-80 mode 1. (Branch to 38H) MSX system
                  software uses interrupt from VDP for timer count.
                  The interval of the interrupt is 60Hz in NTSC and
                  50Hz in PAL/SECAM version.
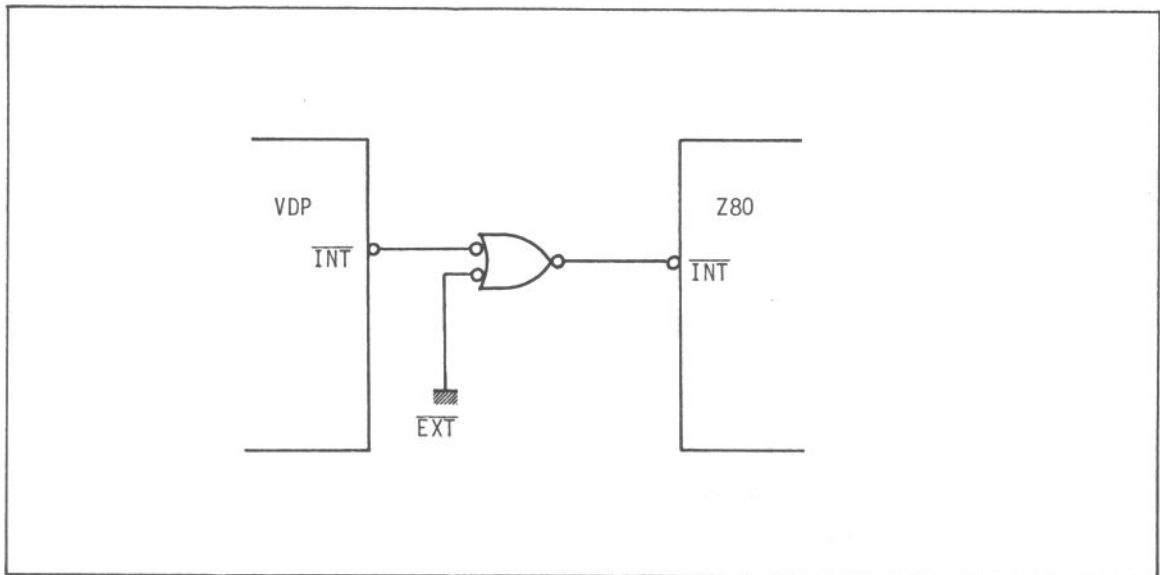
Fig. II-1.

Note: It is not possible to support NMI under MSX-DOS environment
because address 66H, which is the entry vector for NMI, is occupied by
the FCB data of the DOS.

2-4. Screen Display

- LSI   TI TMS9918A Compatible

- Character set   Alphanumerical + European + Graphic
                  256 patterns 6 x 8 dots

- Colour   16 colours

- Sprites   32 sprites. Maximum 4 sprites on the same horizontal
                  line.

MODE		RES.	SIZE	NO Δ	COLOUR	SPRIT	NUMBER OF CHARACTERS
Graphic I	LSI Spec.	256 x 192	8 x 8	256	16 colours	YES	32 x 24
	Suggested Value	240 x 192					29 x 24
Graphic II	LSI Spec.	256 x 192	8 x 8	768	16 colours	YES	32 x 24
	Suggested Value	240 x 192					29 x 24
Multi-colour	LSI Spec.	64 x 48blk	4 x 4 /block	–	16 colours	NO	32 x 24
	Suggested Value	64 x 40blk					29 x 24
Text	LSI Spec.	256 x 192	6 x 8	256	2 colours out of 16 colours	YES	40 x 24
	Suggested Value	240 x 192					39 x 24

Table II-1. List of Display Modes

Suggested value to use: the 8 pixels from left and right of horizontal are not used by software.

Δ The number of patterns

2-5. Keyboard

- Layout   Refer to figure
                    U.S./European
                    French
                    German

- Scanning   Software scanning driven by VDP interrupt

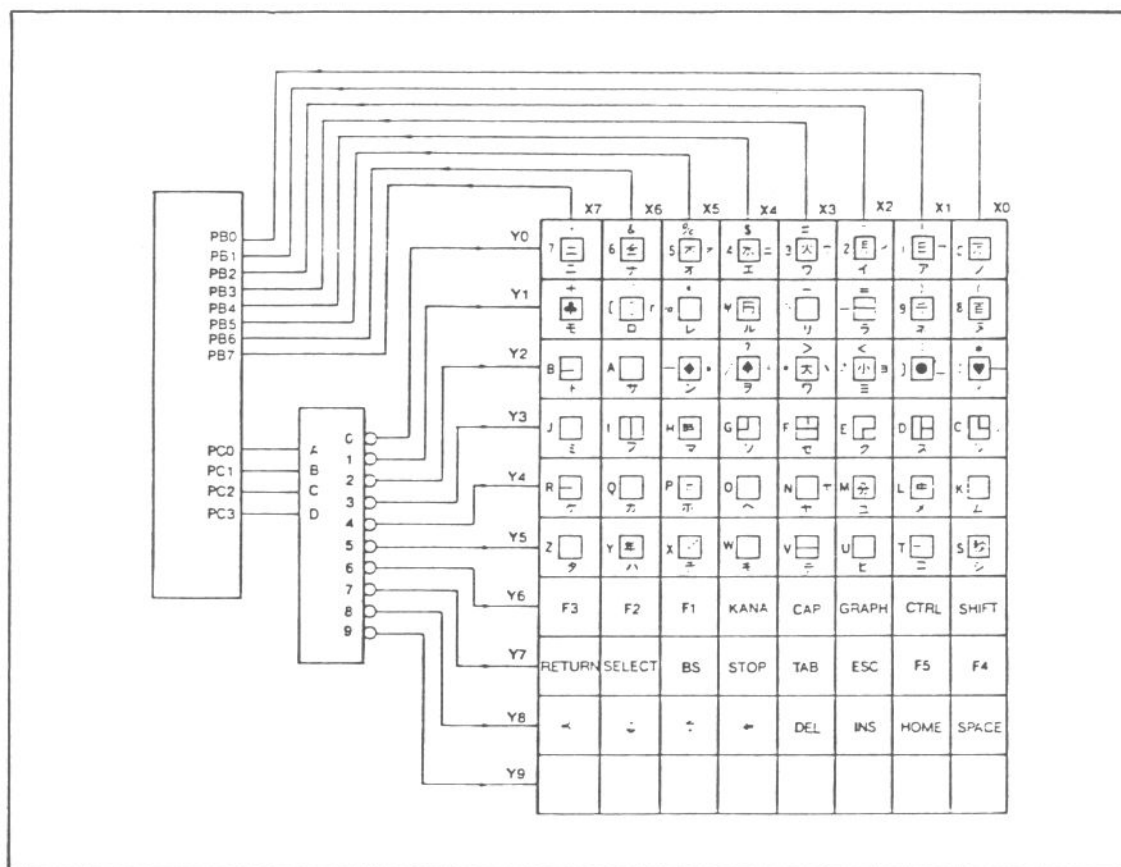- Number of keys   70 plus optional dead key
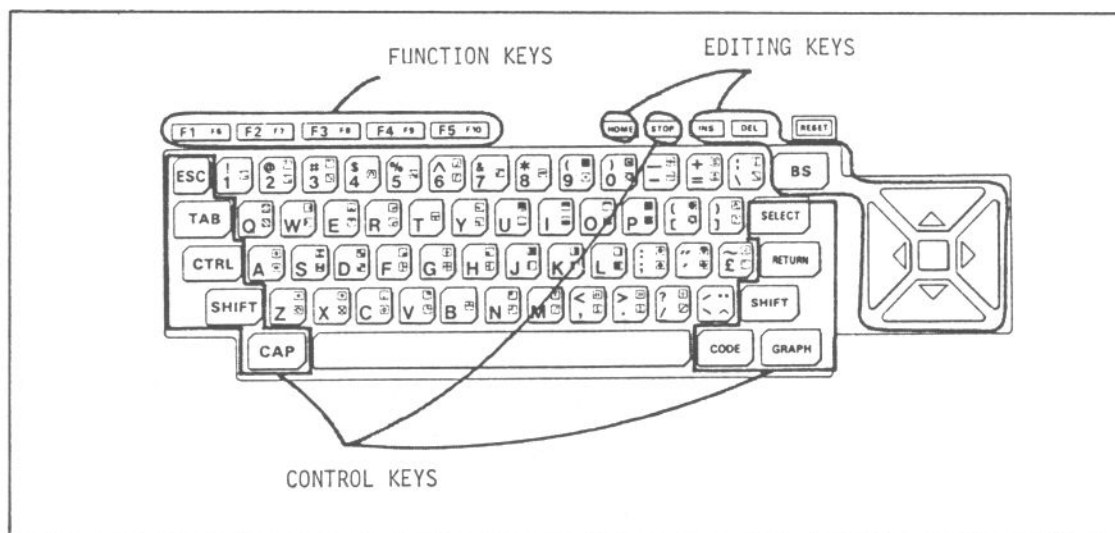
Fig. II-2. Matrix Diagram

Fig. II-3. Keyboard Layout

The following keyboard diagrams contain an optional dead-key. This dead-key is useful for European accented character input. For example, when one wishes to enter "â", one must first press the dead-key "^" and then press "a".

The dead-key will not be useful in the U.S. or U.K. Nor will it be useful in France and Germany, where specific French and German keyboards have been designed which include different dead-keys. Therefore this general dead-key should only be included on machines which will be marketed into minor European countries.

The keyboard diagrams show the dead-key to the left of the carriage return key, but this is probably not a good place for it, because it pushes the carriage return key too far to the right. Manufacturers may place this key where they wish.

2-6. Sound

- LSI	GI AY-3-8910 Compatible
- Octave	Octaves (3 Voices output)
- Sound Effect	Yes
- Software Sound Output	1 bit from output port
- Output Level	-5dBm (If the system has output connector)
- Connector	RCA·2 pins (If the system has audio output connector)

2-7. Cassette Interface

- Input	From the earphone terminal of tape recorder
- Output	To the microphone terminal of tape recorder
- Synchronization	Asynchronous by the software
- Baud Rate	1200 Baud (1200Hz - 1 wave "0", 2400Hz - 2 waves "1") ·(Default) 2400 Baud (2400Hz - 1 wave "0", 4800Hz - 2 waves "1") Change by software (Tape recorder may have to be specified by the manufacturer when used under 2400 Baud mode
- Modulation	FSK (Frequency Shift Keying) by the software

- Demodulation	By the software. The system software auto-matically detects the baud rate when receiving the data.		
- Motor Control	Yes		
- Connector	DIN45326 (8 pin)		
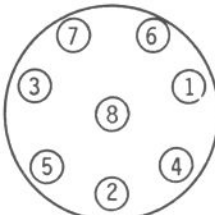
PIN NO	SIGNAL NAME	DIRECTION	PIN CONNECTION
1	GND	---	
2	GND	---	
3	GND	---	
4	CMTOUT	OUTPUT	
5	CMTIN	INPUT	
6	REMOTE +	OUTPUT	
7	REMOTE -	OUTPUT	
8	GND	---	

Table II-2: Signal Pins

2-8. Input/Output (Joystick) Port (1 or 2* ports)

- LSI	AY-3-8910 compatible
- I/O	Input 4 bit, output 1 bit, bidirectional 2 bit per each port
- Logic	Active high
- Level	TTL
- Connector	AMP 9 pin compatible
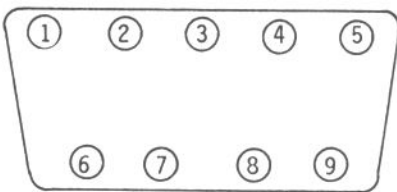
PIN NO.	SIGNAL NAME	DIRECTION	PIN CONNECTION
1	FWD	INPUT	
2	BACK	INPUT	
3	LEFT	INPUT	
4	RIGHT	INPUT	
5	$+5V^{\Delta}$	---	
6	TRG1	INPUT/	
7	TRG2	OUTPUT	
8	OUTPUT	OUTPUT	
9	GND	---	

Table II-3. List of pins
Δ Current capacity is 50mA each

2-9. *Printer Interface

- Specification   8 bit parallel. Handshakes by BUSY and STROBE signal.

- Level   TTL

- Character Code   Same as MSX display code

- Connector   Amp 14 pin compatible

PIN NO	SIGNAL NAME	PIN CONNECTION
1	PSTB	
2	PDB0	
3	PDB1	
4	PDB2	
5	PDB3	
6	PDB4	
7	PDB5	
8	PDB6	
9	PDB7	
10	N.C.	
11	BUSY	
12	N.C.	
13	N.C.	
14	GND	

Pin connection diagram: top row 7 6 5 4 3 2 1; bottom row 14 13 12 11 10 9 8.

Table II-4. List of Pins

2-10. *Floppy Disk Interface

- Contains 16K bytes of ROM at 4000H that includes:

> *MSX-DOS KERNEL
> *MSX DISK BASIC
> *PHYSICAL DISK I/O DRIVER (Supplied by each manufacturer)

- The hardware interface is not specified. The physical disk I/O driver supplied by the manufacturer should virtualize hardware differences.

- It is desirable to have a mechanism in the disk drive to detect whether the drive door has been opened. It reduces disk accesses which check for disk changes.

- Floppy format is MS-DOS compatible

    3.5 inch micro/FD   512 byte/sector

3. Cartridge

3-1. Physical Cartridge Specification

The internal specification of standard size cartridge
(all measurements in mm)

Front

Connector placement

69.4

109mm

16.8

Specification of extended size cartridge

44*
Minimum
height

109   +0.7
      -0

* No standard will be enforced beyond the above minimum height
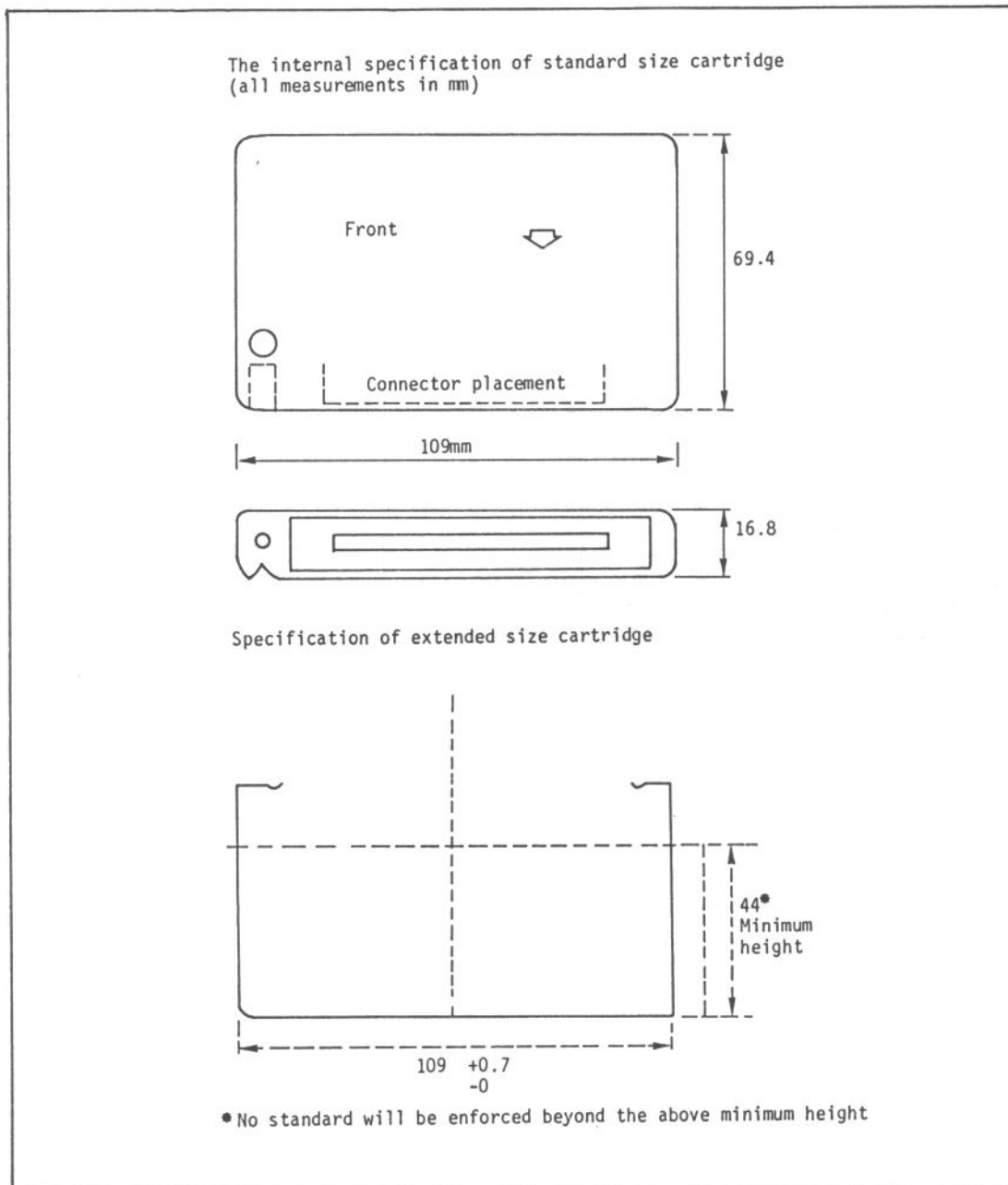
Fig. II-4.

3-2. Cartridge Bus

PIN NO	NAME	I/O*	PIN NO	NAME	I/O*
1	$\overline{CS1}$	O	2	$\overline{CS2}$	O
3	$\overline{CS12}$	O	4	$\overline{SLTSL}$	O
5	RESERVED	–	6	$\overline{RFSH}$	O
7	$\overline{WAIT}$△	I	8	$\overline{INT}$△	I
9	$\overline{M1}$	O	10	$\overline{BUSDIR}$	I
11	$\overline{IORQ}$	O	12	$\overline{MERQ}$	O
13	$\overline{WR}$	O	14	$\overline{RD}$	O
15	$\overline{RESET}$	O	16	RESERVED	–
17	A9	O	18	A15	O
19	A11	O	20	A10	O
21	A7	O	22	A6	O
23	A12	O	24	A8	O
25	A14	O	26	A13	O
27	A1	O	28	A0	O
29	A3	O	30	A2	O
31	A5	O	32	A4	O
33	D1	I/O	34	D0	I/O
35	D3	I/O	36	D2	I/O
37	D5	I/O	38	D4	I/O
39	D7	I/O	40	D6	I/O
41	GND	–	42	CLOCK	O
43	GND	–	44	SW1	–
45	+5V	–	46	SW2	–
47	+5V	–	48	+12V	–
49	SOUNDIN	I	50	–12V	–

Table II-5. List of Signal Pins

* The direction of Input/Output is based on basic unit side.
  Reserved pins must not be used.

△ Open Collector output

PIN NO.	NAME	DESCRIPTION
1	$\overline{CS1}$ Δ	ROM 4000-7FFF selected signal
2	$\overline{CS2}$ Δ	ROM 8000-FFFF selected signal
3	$\overline{CS12}$ Δ	ROM 4000-BFFF selected signal (for 256K ROM)
4	SLTSL	Slot select signal
5	RESERVED	For future use only. Do not use this pin.
6	RFSH	Refresh signal
7	$\overline{WAIT}$	Wait signal to CPU
8	$\overline{INT}$	Interrupt request signal
9	$\overline{M1}$	Fetch cycle signal of CPU
10	$\overline{BUSDIR}$	This signal controls the direction of external data bus buffer when the cartridge is selected. It is low level when data is sent by the cartridge.
11	$\overline{IORQ}$	I/O request signal
12	$\overline{MERQ}$	Memory Request Signal
13	$\overline{WR}$	Write signal
14	$\overline{RD}$	Read signal
15	$\overline{RESET}$	For future use only. Do not use this pin.
17-32	A0-A15	Address bus
33-40	D0-D7	Data bus
41	GND	Ground
42	CLOCK	CPU clock 3.579MHz
43	GND	Ground
44, 46	SW1, SW2	Insert/remove detection for protection
45, 47	+5V	+5V power supply
48	+12V	+12V power supply
49	SOUNDIN	Sound input (-5dBm)
50	-12V	-12V Power supply

Table II-6. Signal Pin Illustration (Underlined signals are negative logic)

CS signals imply memory request and read signal. Therefore they cannot be used as chip select for writable devices such as RAMs.

3-3. Condition of Cartridge Connection

- Fan-in, Fan-out, (LS-TTL load)
  Data and Address bus
  Basic unit side <--------   --------> cartridge side

                     below 2   above 5
     (fan-in)   <--------   <-------- (fan-out)

                     above 1/slot   below 1
     (fan-out   -------->   --------> (fan-in)

- Control signals

                     above 2/slot   below 2
                    -------->   -------->

- Voltage level   TTL level

3-4. Power Capacity

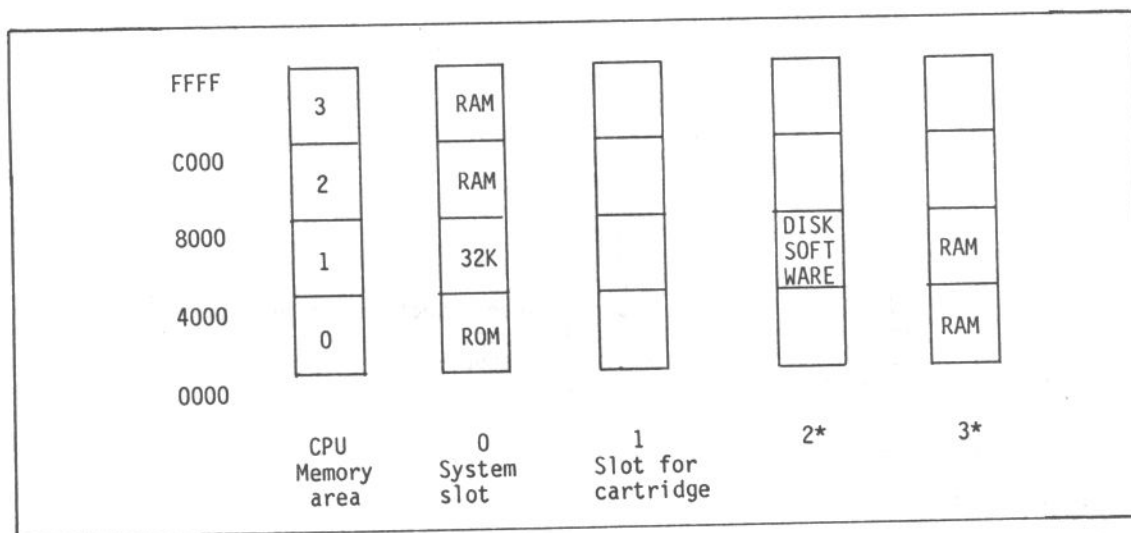|	
+5V	300mA/slot
+12V	50mA
-12V	50mA

4. Memory

4-1. Memory Map

Fig. II-5. Memory Map Example

- MSX BASIC uses the largest available contiguous RAM area that is installed from FFFF to 8000 for its system working RAM area. This can be placed in any slots including expansion slots.

- Slot select register, which is port A of 8255, maps the physical memory space to the logical CPU memory space in 16K byte units (pages). For example, the following value in the slot select register allocates page 0 and 1 from slot 0, page 2 from slot 2 and page 3 from slot 0.
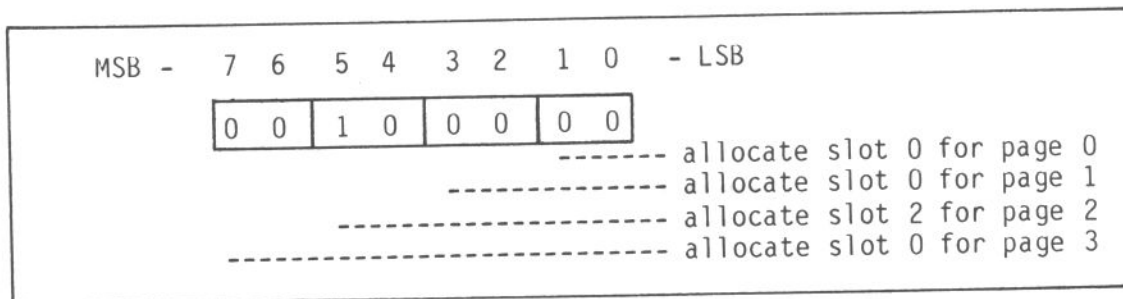

Fig. II-6.

Physical memory is always allocated to the same memory page in the CPU address space. It is not possible to allocate to a different page, like page 3 of slot 3, to page 0 of CPU memory space.

- Minimum system must have two slots, one for system, the other for cartridge.

NOTE:

The word "slot" does not imply that it must have a connector for cartridges, however, a slot for cartridges must have a connector, of course. Refer to APPENDIX C.

- MSX-DOS requires 64K RAM.

4-2. I/O Address Map

FF	
F8	
F7	Audio/Video Control
F0	
E0	
D8	* Kanji character ROM
D0	Δ Floppy disk controller
C0	
B8	* Light Pen interface
B4	
B0	External Memory
A8	PPI (8255)
A0	PSG (AY-3-8910)
98	VDP (9918A)
90	* Printer interface
88	
80	* RS-232C interface
00	Not specified

Fig. II-7. I/O Address Map

* Items with asterisk may not be built-in in the minimum system.

5. I/O Device

5-1. I/O Device Description

5-1-1. RS-232C

5-1-1-1. LSI Components

  i-8251 Communication interface chip
  i-8253 Programmable interval timer chip

5-1-1-2. Port Address

```
80H R/W 8251 data port
81H R/W 8251 command/status port
82H  R  Baud rate setting switches
83H  R  Configuration setting switches
83H  W  Interrupt mask register
84H R/W 8253 counter 0
85H R/W 8253 counter 1
86H R/W 8253 counter 2
87H  W  8253 mode register
```

5-1-1-3. The Usage of Switch Port at Address 82H and 83H

  82H read - Baud rate select

        bit 0 - 3 : baud rate for receiver
        bit 4 - 7 : baud rate for transmitter

VALUE	BAUD RATE
0	50
1	75
2	110
3	150
4	300
5	600
6	1200
7	2400
8	4800
9	9600
A	19200
B	N.A.
C	N.A.
D	N.A.
E	N.A.
F	disable *

* When value F is set as a baud rate, that function is disabled by
  software.

- 18 -

83H read - Set various function

```
        bit 0 - CD (carrier detect)*
            1 - Auto line feed on receive**    1 - Auto line feed
            2 - Full/Half duplex               1 - Full duplex
            3 - XON/OFF control                1 - Enable control
            4 - Word length                    1 - 8 bits, 0 - 7 bits
            5 - Parity Even/Off                1 - Even
            6 - Parity enable                  1 - Enable
            7 - Stop bit length                1 - 2 bits, 0 - 1 bit
```

*   CD is a signal directly connected to carrier detect (pin 8) on the
    DB-25 connector.

** Add line feed on receiving carriage return.

NOTE:

Bit 0 of the switch pulls up the CTS line of the 8251 (or [actually] it
pulls down since CTS on 8251 is negative logic) to make it possible to
send data even when CTS is not supplied from outside.

83H write - Set interrupt mask for receive

bit 0 -   mask interrupt for receive; 1 - mask interrupt
              The initial value of this mask is 1 (disable interrupt).

5-1-1-4. Usage of 8253 Timer-Counter To Generate Baud Rate -
         Clock for 8251

- Frequency of crystal
              The frequency of the crystal:
                      1.2288MHz

- Usage of counter channel
              CH0 - Rx baud rate clock
              CH1 - Tx baud rater clock
              CH2 - General interrupt timer ... connect to IRQ

5-1-2. Printer Port

5-1-2-1. Port Address

   90H R Busy status : bit 1
   90H W Strobe output : bit 0
   91H W Print data

- 19 -

5-1-3. VDP Port

98H R/W Video Ram Data
99H R/W Command and status register

5-1-4. PSG Port

A0H W Address latch
A1H W Data write
A2H R Data read

.

5-1-5. PPI Port

A8H R/W Port A
A9H R/W Port B
AAH R/W Port C
ABH R/W Mode register

5-1-6. External Memory (Sony)

B0H through B3H

5-1-7. Light Pen (Sanyo)

B8H through BBH

5-1-8. Audio/Visual Control

F7H W BIT4 - AV CONTROL   L - TV
    W BIT5 - Ym CONTROL   L - TV
    W BIT6 - Ys CONTROL   L - Super
    W BIT7 - Video select   L - TV

5-2. Notes on I/O Address Assignment

- I/O address 80 ç FF are assigned for system usage. The empty areas
  are reserved for system use.
  Although these addresses are defined here, software should not access
  those devices directly through the addresses listed above. Every
  access to the I/O must be done through the BIOS calls. This is to
  keep software independent from hardware differences. Manufacturers
  may change some hardware from the standard MSX system but still be
  able to maintain software compatibility by supporting the hardware

differences within the BIOS, so that the difference can be transparent to software.

The only exception is access to the VDP. Locations 6 and 7 of the MSX system ROM contain read and write addresses of VDP register. The software needing to access VDP very quickly may access VDP directly through those addresses stored in ROM.

- 00 ç 7F are free addresses, however when different devices use the same address, they may not be accessed at the same time. Basically, special I/O devices not defined here should be placed in the memory space as memory-mapped I/O. Refer to Appendix B.3.

FDC may be placed in I/O space, but it must have a mechanism to disable it and only at the moment when the system accesses the FDC, is it enabled. This makes it possible to have more than one FDC interface in the system to handle different kinds of media.

5-3. 8255 (PPI) Bit Assignment

PORT	BIT	I/O	SIGNAL NAME	DESCRIPTION
A	0 1	O U T P U T	CS0L SS0H	0000~3FFF address slot select signal
	2 3		CS1L CS1H	4000~7FFF address slot select signal
	4 5		CS2L CS2H	8000~BFFF address slot select signal
	6 7		CS3L CS3H	C000~FFFF address slot select signal
B	0 thru 7	INPUT		Keyboard return signal
C	0 1 2 3	O U T P U T	KB0 KB1 KB2 KB3	Keyboard scan signal
	4		CASON	Cassette control signal (L-ON)
	5		CASW	Cassette write signal
	6		CAPS	CAPS lamp signal("L" --> ON)
	7		SOUND	Sound input by software

Table II-7. 8255 (PPI) Bit Assignment

PORT	BIT	I/O	CONNECTOR PIN NO.		NOTE
A	0		J3-PIN 1	1	FWD1
			J4-PIN 1*	2	FWD2
	1		J3-PIN 2	1	BACK1
			J4-PIN 2*	2	BACK2
	2	I	J3-PIN 3	1	LEFT1
		N	J4-PIN 3*	2	LEFT2
	3	P	J3-PIN 4	1	RIGHT1
		U	J4-PIN 4*	2	RIGHT2
	4	T	J3-PIN 6	1	TRGA1
			J4-PIN 6*	2	TRGA2
	5		J3-PIN 7	1	TRGB1
			J4-PIN 7*	2	TRGB2
	6		KEY LAYOUT Select 4		Japanese version only
	7		CSAR (CASSETTE TAPE READ		
B	0		J3-PIN 6	3	--- "H" LEVEL
	1	O	J3-PIN 7*	3	--- "H" LEVEL
	2	U	J4-PIN 6	3	--- "H" LEVEL
	3	T	J4-PIN 7*	3	--- "H" LEVEL
	4	P	J3-PIN 8		
	5	U	J3-PIN 8*		
	6	T	PORT A INPUT SELECT		Selects J3/J4
	7		KLAMP (KANA LAMP L-ON)		Japanese version only

Table II-8. PSG Bit Assignment

1 Available when bit 6 of port B is low used by JOYSTICK 1
2 Available when bit 6 of port B is high used by JOYSTICK 2
3 Turn to "H" level when using those pins as an input port.
  Tied an open collector buffer to the output. (Refer to Appendix C-1)
4 JIS layout - "H" level, syllable layout - "L" level

  Remark   PIN 5 + 5V
           PIN 9 GND

- On the minimum system, there is no J4 connector.

APPENDIX A

LIST OF CONNECTORS

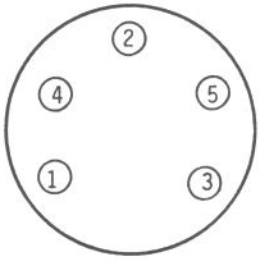| PIN NAME	SPECIFICATION
1. Video output and composite video	DIN 5 PIN CONNECTOR Δ or RCA 2 PIN CONNECTOR
2. RF modulated signal	RCA 2 PIN CONNECTOR
CASSETTE	DIN 8 PIN CONNECTOR (DIN-45326)
I/O PORT	AMP 9 PIN CONNECTOR
PRINTER	AMPHENOL 14 PIN CONNECTOR
CARTRIDGE BUS	0.10 inch (2.54mm) SPACE, 50 PIN CONNECTOR
AUDIO	RCA 2 PIN CONNECTOR

Table II-9.

Δ DIN 5 PIN Connector Signal Pin Assignment

PIN NO.	NAME	PIN CONNECTION
1	+5V	
2	GND	
3	AUDIO	
4	MONITOR	
5	RF VIDEO	

Table II-10. List of Connectors

APPENDIX B

NOTES ON SYSTEM EXPANSION

B-1. RAM Expansion

- The MSX BASIC needs contiguous RAM space starting from FFFF down to 8000. Therefore, additional RAM should be added to existing RAM to form contiguous RAM space.

- If the basic model has only 8K bytes of RAM, it is not possible to add RAM in the slot other than the original 8K in place, because the slot select logic treats memory space in 16K byte units. So an expansion cartridge that has 16K bytes adds only 8K bytes when it is placed at location C000 to FFFF. It is not possible to expand the RAM area by placing it at location 8000 to BFFF because there would be no RAM from C000 to DFFF.

- The BASIC MSX software only uses RAM from 8000 to FFFF, so the RAM installed from 0 to 7FFF cannot be used by it.

B-2. Slot Expansion

- To expand slots, the additional slots are expanded from a primary slot. Primary slots are those slots which are managed by the basic slot select register placed in port A of 8255. Therefore, to select the expanded slot, first select the primary slot to which the expanded slot is connected, and select the desired expansion slot.

- The location of the slot select register for the expanded slots is at memory address FFFF of the primary slot. To make it possible to differentiate this register from ordinary RAM, complement the output of the register. That is, when reading the register, the read data is the complement of the actual value of the register.

- The maximum number of cartridges that can be connected to the cartridge bus is four. Therefore, buffers are necessary to put more than five slots to the system. The control signal that controls the direction of those buffers is BUSDIR. Those devices which send signals to the CPU and are placed in an expanded slot have to send BUSDIR signal as well the change in the direction of the buffer from expanded slots to CPU.
However, for the accesses to memory, it is possible to know the direction of the bus using the slot select signal to the primary slot, memory request signal and read/write signal. So the direction of the buffer should be controlled around the buffer circuitry. So, those cartridges which contain only ROM/RAM do not have to manage BUSDIR signal, which makes them cheap. But those cartridges that contain

- 25 -

some devices which send signals to the CPU, that is, those devices that respond to INP instruction or those devices that are responsible to supply the address in response to the mode 2 of interrupt, must force BUSDIR at "L" level when they send data to the CPU.

B-3. I/O Expansion

- In the Z-80 based system, it is common to place I/O devices in I/O address space. But the MSX sysem design is so flexible and expandable, it is possible to add some I/O devices with cartridge that share same address space. If this is the case, none of those devices can be accessed properly.
To avoid this situation, it is preferred to place I/O devices in the memory spaces because they are managed by slot select logic and no memory can be accessed simultaneously when they are placed in different slots. But the devices placed in memory space cannot be accessed by the softwares that run in different slots. So those devices that are general to any software such as VDP have to be placed in I/O address space. Also in some case, it is cheaper to use I/O address space because only 8 bits address information should be decoded. I/O address space from 80 to FF is defined for those system devices. But the addresses below 7F are left free.

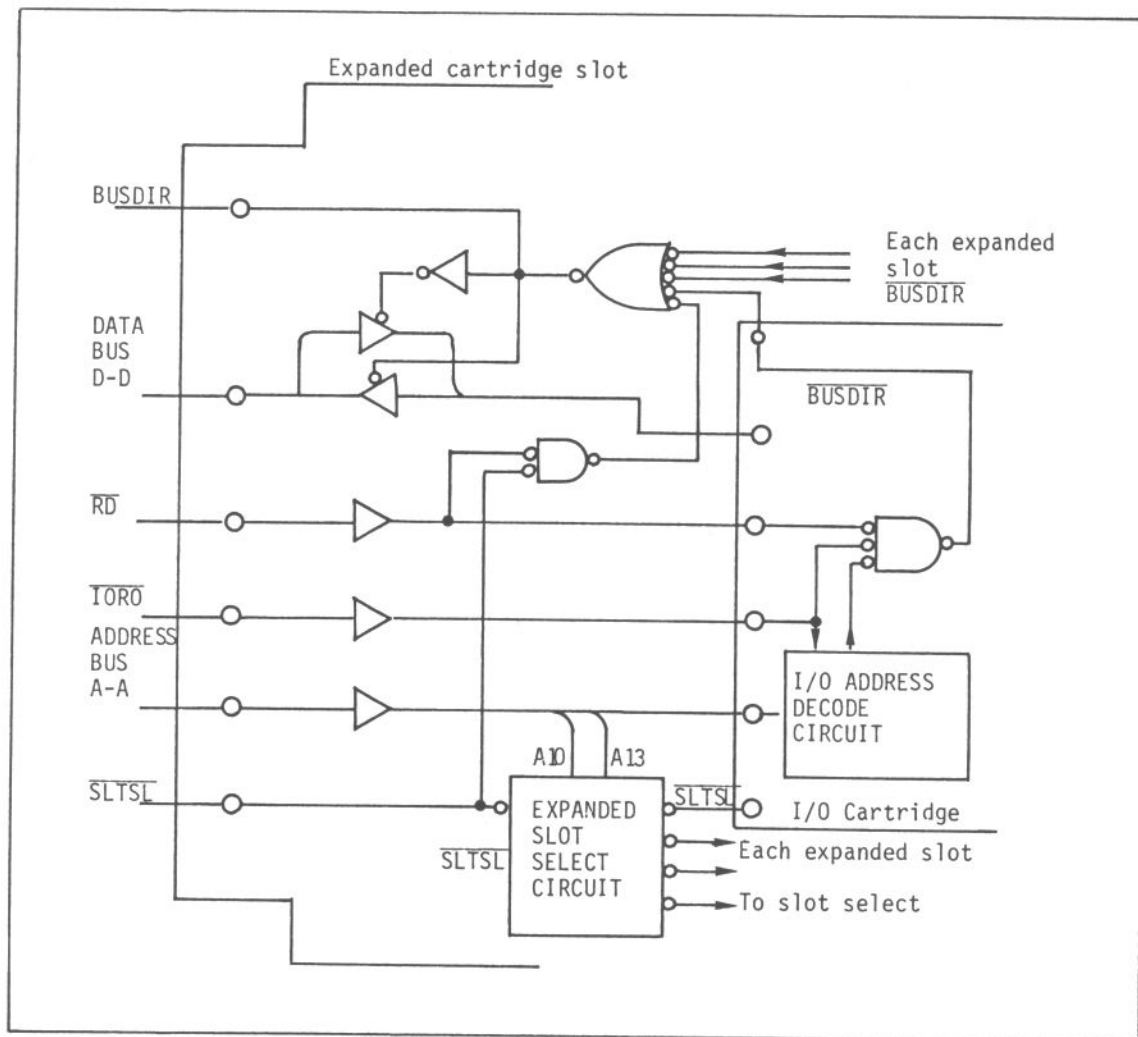## B-4. Sample Circuit of Expanded Slot

Fig. II-8.

B-5. Slot

- Concept of Slot

  For a structure of 64 KB memory space, the concept of slot and memory bank is nearly the same. But CPU can choose the cartidge by the slot number in which it is inserted.

  The slot concept is originated from the standpoint of the software, and so, the software is independent to the number of slot that has actual slot opening.

- Advantage of Slot Structure

  In the common bus structure, when the memory bank number is even, the device select signal, connected to the bus, cannot distinguish different devices if they use the same memory area, in this case, not only the system becomes out of order, hardware will suffer from deterioration. By using slot select signal to choose each of the device, there will be no such problem, program dueing with two or more devices at the same memory area, can be possible. This is favourable to the flexibility and expandability of the system.
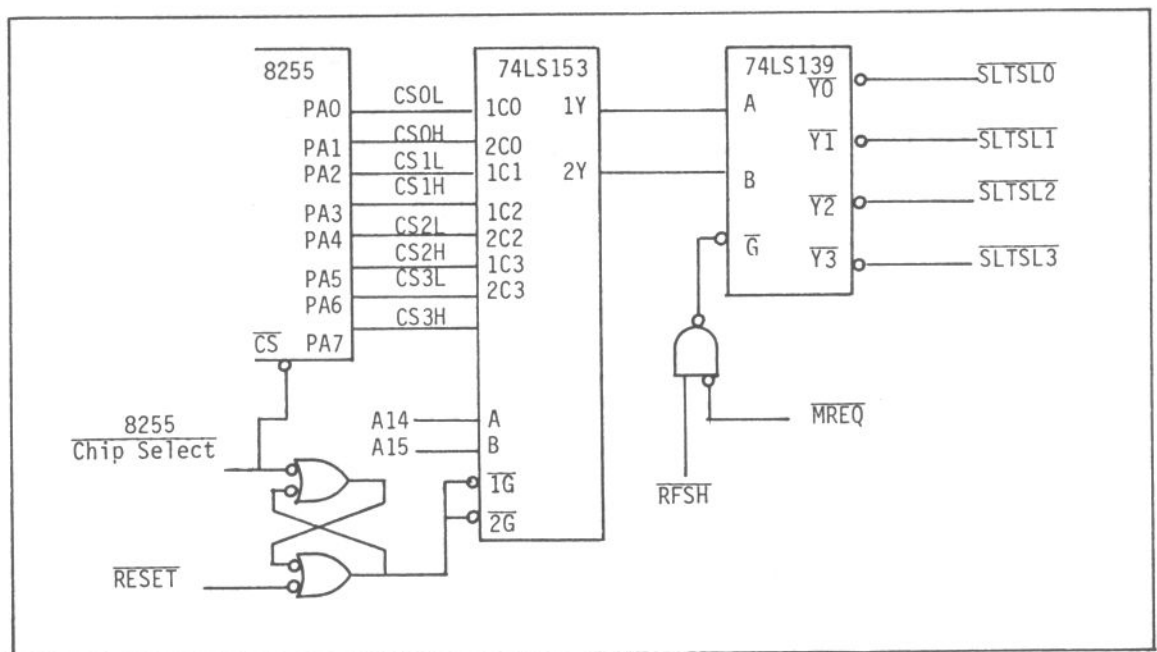

Fig. II-9. Slot Select Circuit Diagram
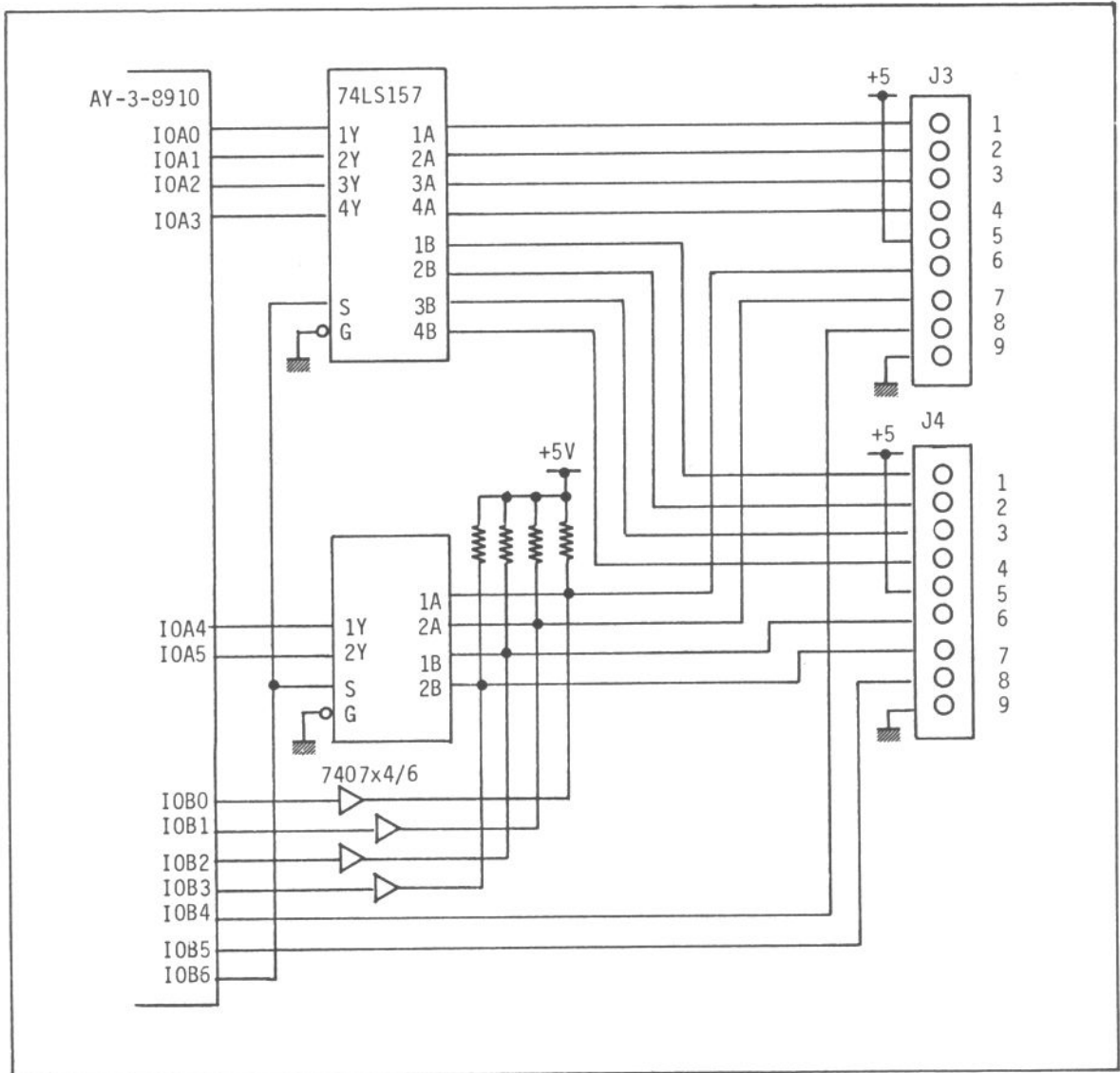
SAMPLE CIRCUIT DIAGRAM OF GENERAL I/O PORT

Fig. II-10. Circuit Diagram of General I/O Port

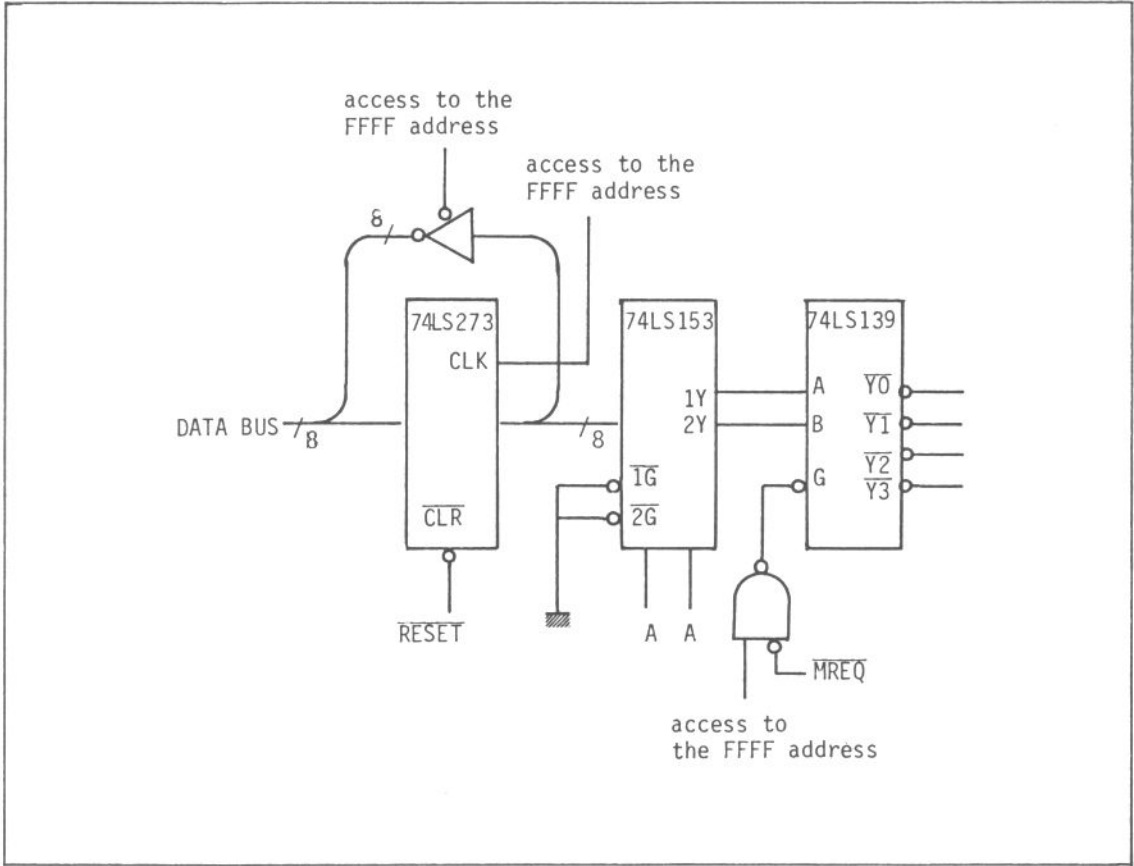## SAMPLE CIRCUIT DIAGRAM OF EXPANDED SLOT SELECT SIGNAL GENERATE LOGIC

Fig. II-11. Circuit Diagram of Expanded Slot Select Signal Generate Logic

THE MSX STANDARD CHIP SET

1. The Z80 CPU

1-1. Features

- The instruction set contains 158 instructions. The 78 instructions of the 8080A are included as a subset; 8080A software compatibility is maintained.

- 8MHz, 6MHz, 4MHz and 2.5MHz clocks for the Z80H, Z80B, Z80A and Z80 CPU result in rapid instruction execution with consequent high data throughput.

- The extensive instruction set includes string, bit, byte and word operations. Block searches and block transfers together with indexed and relative addressing result in the most powerful data handling capabilities in the microcomputer industry.
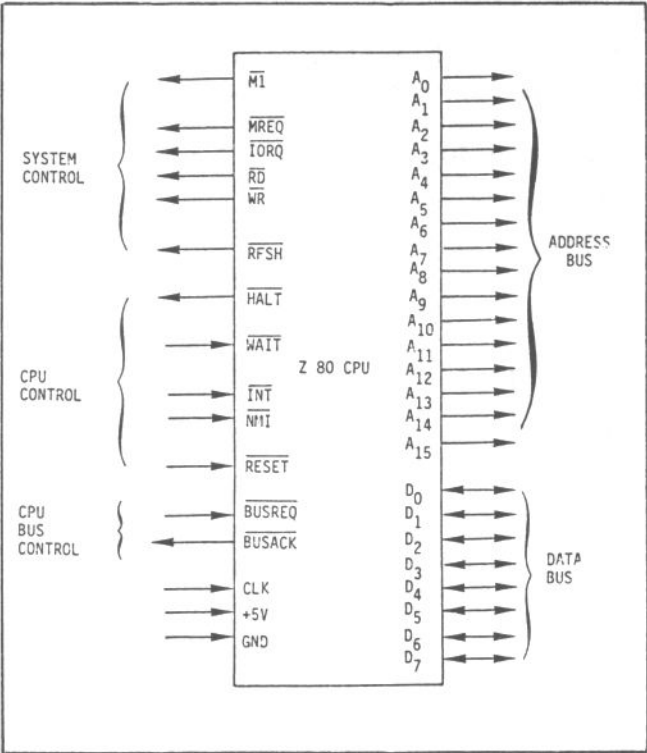
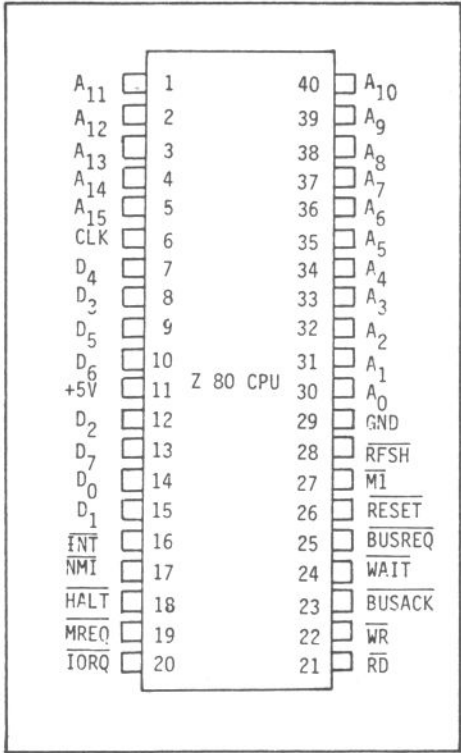Fig. III-1. Pin Functions   Fig. III-2. Pin Assignments

- The Z80 microprocessors and associated family of peripheral controllers are linked by a vectored interrupt system. This system may be daisy-chained to allow implementation of a priority interrupt scheme. Little, if any, additional logic is required for daisy-chaining.

- Duplicate sets of both general-purpose and flag registers are provided, easing the design and operation of system software through single-context switching, background-foreground programming, and single-level interrupt processing. In addition, two 16-bit index registers facilitate program processing of tables and arrays.

- There are three modes of high-speed interrupt processing: 8080 similar, non-Z80 Family peripheral with or without daisy chain.

- On-chip dynamic memory refresh counter.

1-2. General Description

The Z80, Z80A, Z80B and Z80H CPU's are third-generation single-chip microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second- and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either sets of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may be reserved for very fast interrupt response.
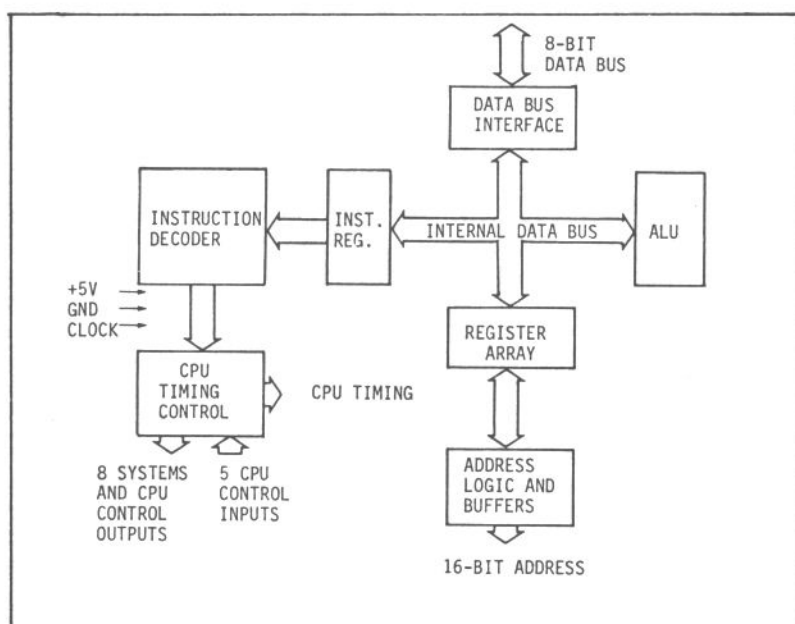

Fig. III-3. Z80 CPU Block Diagram

The Z80 also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register. The CPU is easy to incorporate into a system since it requires only a single +5V power source. All output signals are fully decoded and timed to control standard memory or peripheral circuits, and it is supported by an extensive family of peripheral controllers. The internal block diagram (Fig. III-3) shows the primary functions of the Z80 processors. Subsequent text provides more detail on the Z80 I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.

1-3. Z80 Microprocessor Family

The Zilog Z80 microprocessor is the central element of a comprehensive microprocessor product family. This family works together in most applications with minimum requirements for additional logic, facilitating the design of efficient and cost-effective microcomputer-based systems.

Zilog has designed five components to provide extensive support for the Z80 microprocessor. These are:

- The PIO (Parallel Input/Output) operates in both data-byte I/O transfer mode (with handshaking) and in bit mode (without handshaking). The PIO may be configured to interface with standard parallel peripheral devices such as printers, tape punchers, and keyboards.

- The CTC (Counter/Timer Circuit) features four programmable 8-bit counter/timers, each of which has an 8-bit prescaler. Each of the four channels may be configured to operate in either counter or timer mode.

- The DMA (Direct Memory Access) controller provides dual port data transfer operations and the ability to terminate data transfer as a result of a pattern match.

- The SIO (Serial Input/Output) controller offers two channels. It is capable of operating in a variety of programmable modes for both synchronous and asynchronous communication, including Bi-Sync and SDLC.

- The DART (Dual Asynchronous Receiver/Transmitter) device provides low cost asynchronous serial communication. It has two channels and a full modem control interface.

1-4. Z80 CPU Registers

Fig. III-4. shows three groups of registers within the Z80 CPU. The first group consists of duplicate sets of 8-bit registers: a principal set and an alternate set (designated by '[prime], e.g., A'). Both sets consist of the Accumulator Register, the Flag Register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques as background-foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt Register), the R (Refresh Register), the IX and IY (Index Registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus an additional pair of flip-flops which assists in identifying the interrupt mode at any particular time.

MAIN REGISTER SET		ALTERNATE REGISTER SET	
A ACCUMULATOR	F FLAG REGISTER	A' ACCUMULATOR	F' FLAG REGISTER
B GENERAL PURPOSE	C GENERAL PURPOSE	B' GENERAL PURPOSE	B' GENERAL PURPOSE
D GENERAL PURPOSE	E GENERAL PURPOSE	D' GENERAL PURPOSE	E' GENERAL PURPOSE
H GENERAL PURPOSE	L GENERAL PURPOSE	H' GENERAL PURPOSE	L' GENERAL PURPOSE

←— 8 BITS —→

←——— 16 BITS ———→   INTERRUPT FLIP-FLOPS STATUS

| INDEX REGISTER | IFF1 | IFF2 |

| IY INDEX REGISTER |

0 = INTERRUPTS DISABLED   STORES IFF1
1 = INTERRUPTS ENABLED   DURING NMI SERVICE

| SP STACK POINTER |

| PC PROGRAM COUNTER | INTERRUPT MODE FLIP-FLOPS

| I INTERRUPT VECTOR | R MEMORY REFRESH |

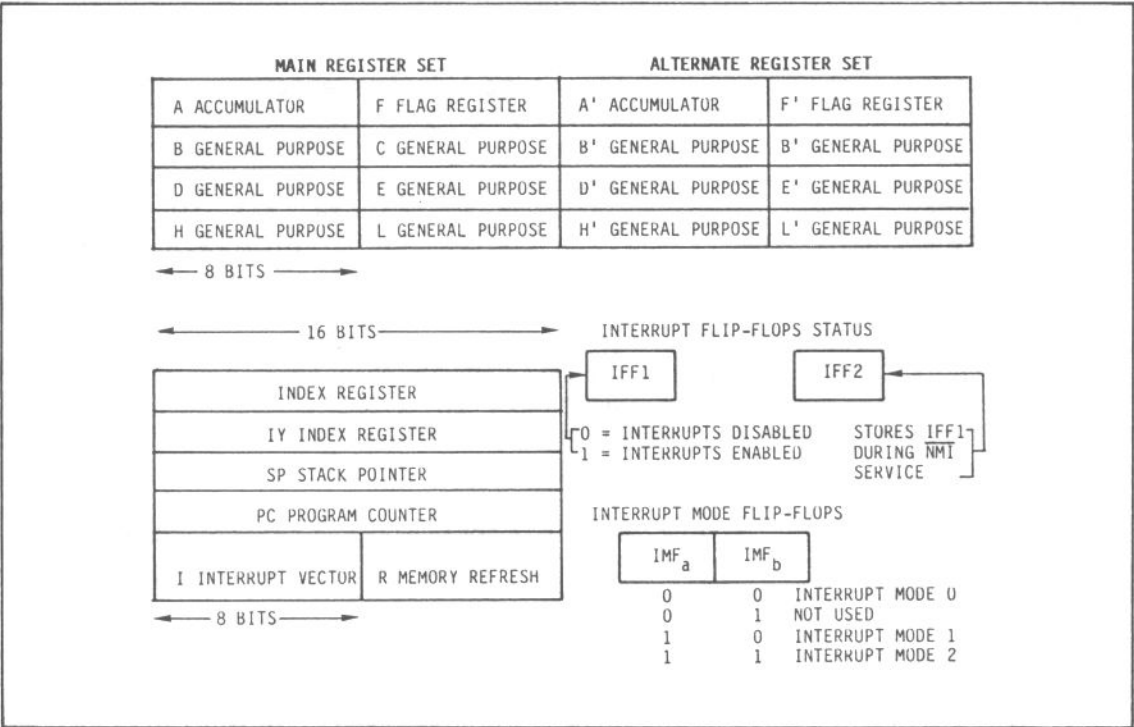| IMF_a	IMF_b	
0	0	INTERRUPT MODE 0
0	1	NOT USED
1	0	INTERRUPT MODE 1
1	1	INTERRUPT MODE 2

←— 8 BITS —→

Fig. III-4. CPU Registers

1-5. Interrupts: General Operation

The CPU accepts two interrupt input signals: NMI and INT. The NMI is a non-maskable interrupt and has the highest priority. INT is a lower priority interrupt and it requires that interrupts be enabled in software in order to operate. INT can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80 has a single response mode for interrupt service for the non-maskable interrupt. The maskable interrupt, INT, has three programmable response modes available.

These are:

- Mode 0 - similar to the 8080 microprocessor
- Mode 1 - Peripheral Interrupt service, for use with non-8080/Z80 systems.
- Mode 2 - A vectored interrupt scheme, usually daisy-chained, for use with Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the NMI and INT signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section

Non-Maskable Interrupt (NMI).
The non-maskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU. NMI is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shut-down after power failure has been detected. After recognition of the NMI signal (providing BUSREQ is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routing.

Maskable Interrupt (INT).
Regardless of the interrupt mode set by the user, the Z80 response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and BUSREQ is not active) a special interrupt processing cycle begins. This is a special fetch (M1) cycle in which IORQ becomes active rather than MREQ, as in normal M1 cycle. In addition, this special M1 cycle is automatically extended by two WAIT states, to allow for the time required to acknowledge the interrupt request.

Mode 0 Interrupt Operation
This mode is similar to the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus. This is normally a Restart instruction, which will initiate a call to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

Mode 1 Interrupt Operation.
Mode 1 operation is very similar to that for the NMI. The principal difference is that the Mode 1 interrupt has a restart location of 0038H only.

Mode 2 Interrupt Operation.
This interrupt mode has been designed to utilize most effectively the capabilities of the Z80 microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8-bits and the contents of

the I register as the upper 8-bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines. These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

Interrupt Priority (Daisy Chaining and Nested Interrupts).
The interrupt priority of each peripheral device is determined by its physical location within a daisy-chain configuration. Each device in the chain has an interrupt enable input line (IEI) and an interrupt enable output line (IEO), which is fed to the next lower priority device. The first device in the daisy chain has its IEI input hardwired to a High level. The first device has highest priority, while each succeeding device has a corresponding lower priority. This arrangement permits the CPU to select the highest priority interrupt from several simultaneously interrupting peripherals.

The interrupting device disables its IEO line to the next lower priority peripheral until it has been serviced. After servicing, its IEO line is raised, allowing lower priority peripherals to demand interrupt servicing.
The Z80 CPU will nest (queue) any pending interrupts or interrupts received while a selected peripheral is being serviced.

Interrupt Enable/Disable Operation.
Two flip-flops, IFF_1 and IFF_2, referred to in the register description are used to signal the CPU interrupt status. For more details, refer to the Z80 CPU Technical Manual and Z80 Assembly Language Manual.

Action	IFF_1	IFF_2	Comments
CPU Reset	0	0	Maskable interrupt $\overline{INT}$ disabled
DI Instruction execution	0	0	Maskable interrupt $\overline{INT}$ disabled
IE instruction execution	1	1	Maskable interrupt $\overline{INT}$ enabled
LD A, I instruction execution	.	.	IFF_2 - Parity flag
LD A ,R instruction execution	.	.	IFF_1 - Parity flag
Accept $\overline{NMI}$	0	IFF_1	IFF_1 - IFF_2 (Maskable interrupt $\overline{INT}$ disabled)
RETN instruction execution	IFF_2	.	IFF_2 - IFF_1 at completion of an $\overline{NMI}$ service routine

Table III-1. State of Flip-Flops

1-6. Instruction Set

The Z80 microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast efficient data transfers within memory or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the Z80 instruction set and shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. The Z80 CPU Technical Manual (03-0029-01) and Assembly Language Programming Manual (03-0002-01) contain significantly more details for programming use.

The instructions are divided into the following categories:

- 8-bit loads
- 16-bit loads
- Exchanges, block transfers, and searches
- 8-bit arithmetic and logic operations
- General-purpose arithmetic and CPU control
- 16-bit arithmetic operations
- Rotates and shifts
- Bit set, reset, and test operations
- Jumps
- Calls, returns, and restarts
- Input and output operations

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include:

- Immediate
- Immediate extended
- Modified page zero
- Relative
- Extended
- Indexed
- Register
- Register indirect
- Implied
- Bit

1-7. Pin Descriptions

$A_0 - A_{15}$.

Address Bus (output, active High, 3-state). $A_0 - A_{15}$ for a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.

BUSACK. Bus Acknowledge (output, active Low).
Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals MREQ, IORQ, RD , and WR have entered their high-impedance states. The external circuitry can now control these lines.

BUSREQ. Bus Request (input, active Low).
Bus Request has a higher priority than NMI and is always recognized at the end of the current machine-cycle. BUSREQ forces the CPU address bus, data bus, and control signals MREQ, IORQ, RD and WR to go to a high-impedance state so that other devices can control these lines. BUSREQ is normally wire-ORed and requires an external pullup for these applications. Extended BUSREQ periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.

D_0 - D_7. Data Bus (input/output, active High, 3-state).

D_0 - D_7 constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

HALT. Halt State (output, active Low).
HALT indicates that the CPU has executed a Halt instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can resume.
While halted, the CPU executes NOPs to maintain memory refresh.

INT. Interrupt Request (input, active Low).
Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software controlled interrupt enable flip-flop (IFF) is enabled. INT is normally wire-ORed and requires an external pullup for these applications.

IORQ. Input/Output Request (output, active Low, 3-state). IORQ indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation.
IORQ is also generated concurrently with M1 during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.

M1. Machine Cycle One (output, active Low).
M1, together with MREQ, indicates that the current machine cycle is the opcode fetch cycle of an instruction execution. M1, together with IORQ, indicates an interrupt acknowledge cycle.

MREQ. Memory Request (output, active Low, 3-state). MREQ indicates that the address bus holds a valid address for a memory read or memory write operation.

NMI. Non-Maskable Interrupt (input, negative edge-triggered).
NMI has a higher priority than INT. NMI is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.

RD. Read (output, active Low, 3-state).
RD indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

RESET. Reset (input, active Low).
RESET initializes the CPU as follows: it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state.
Note that RESET must be active for a minimum of three full clock cycles before the reset operation is complete.

RFSH. Refresh (output, active Low).
RFSH, together with MREQ, indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.

WAIT. Wait (input, active Low).
WAIT indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended WAIT periods can prevent the CPU from refreshing dynamic memory properly.

WR. Write (output, active low, 3-state).
WR indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

1-8. CPU Timing

The Z80 CPU executes instructions by proceeding through a specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a T time of cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by insertion of one or more Wait states by the user.

2. The 8255 PPI

2-1. Description

This is a family of general-purpose programmable input/output devices designed for use with the M5L 8085A 8-bit parallel CPU as input/output ports. These devices are fabricated using N-channel silicon-gate ED-MOS technology for a single supply voltage. They are simple input and output interfaces for TTL circuits, having 24 input/output pins which correspond to three 8-bit input/output ports.

2-2. Features

- 24 programmable I/O pins

- Single 5V supply voltage

- TTL-compatible I_{OL} = 2.5mA (max)

- Fully compatible with MELPS 8 microprocessor series

- Direct bit set/reset capability

- Interchangeable with Intel's 8255A in terms of function, electrical characteristics and pin configuration.
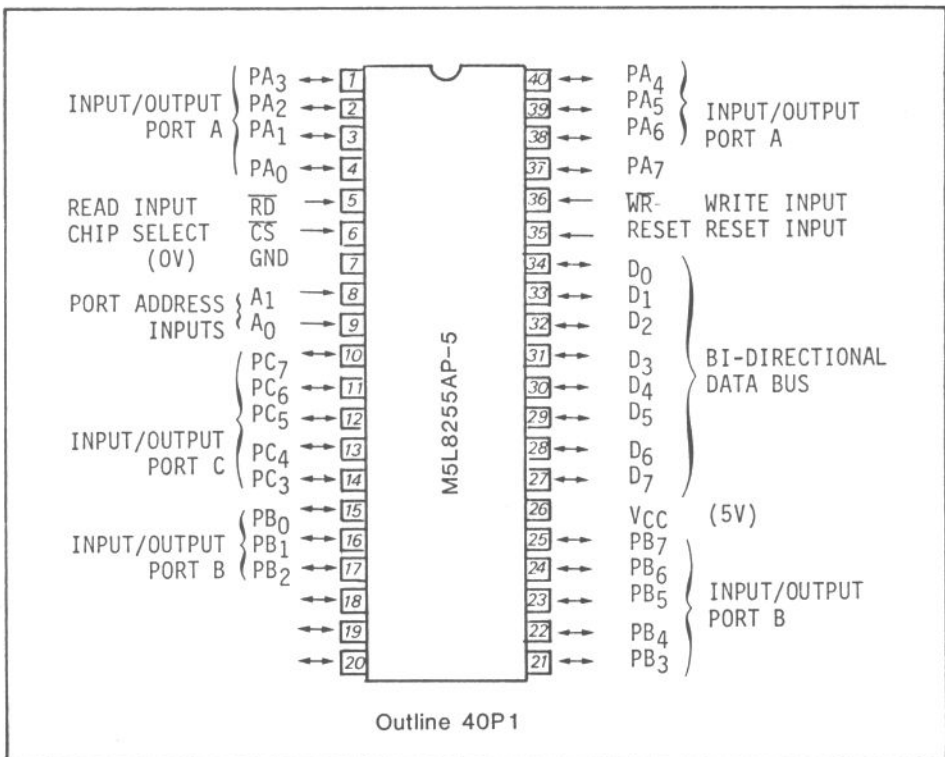
Fig. III-5. Pin Configuration

2-3. Application

- Input/output for MELPS 8/85 microprocessor

2-4. Function

These PPIs have 24 input/output pins which may be individually programmed in two 12-bit groups A and B with mode control commands from a CPU. They are used in three major modes of operation, mode 0, mode 1 and mode 2.

Operating in mode 0, each group of 12 pins may be programmed in sets of 4 to be inputs or outputs. In mode 1, the 24 I/O terminals may be programmed in two 12-bit groups, group A and B. Each group contains one 8-bit data port, which may be programmed to serve as input or output, and one 4-bit control port used for handshaking and interrupt control signals. Mode 2 is used with group A only, as one 8-bit bidirectional bus port and one 5-bit control port.

Bit set/reset is controlled by CPU. A high-level reset input (RESET) clears all internal registers, and all ports are set to the input mode (high-impedance state).
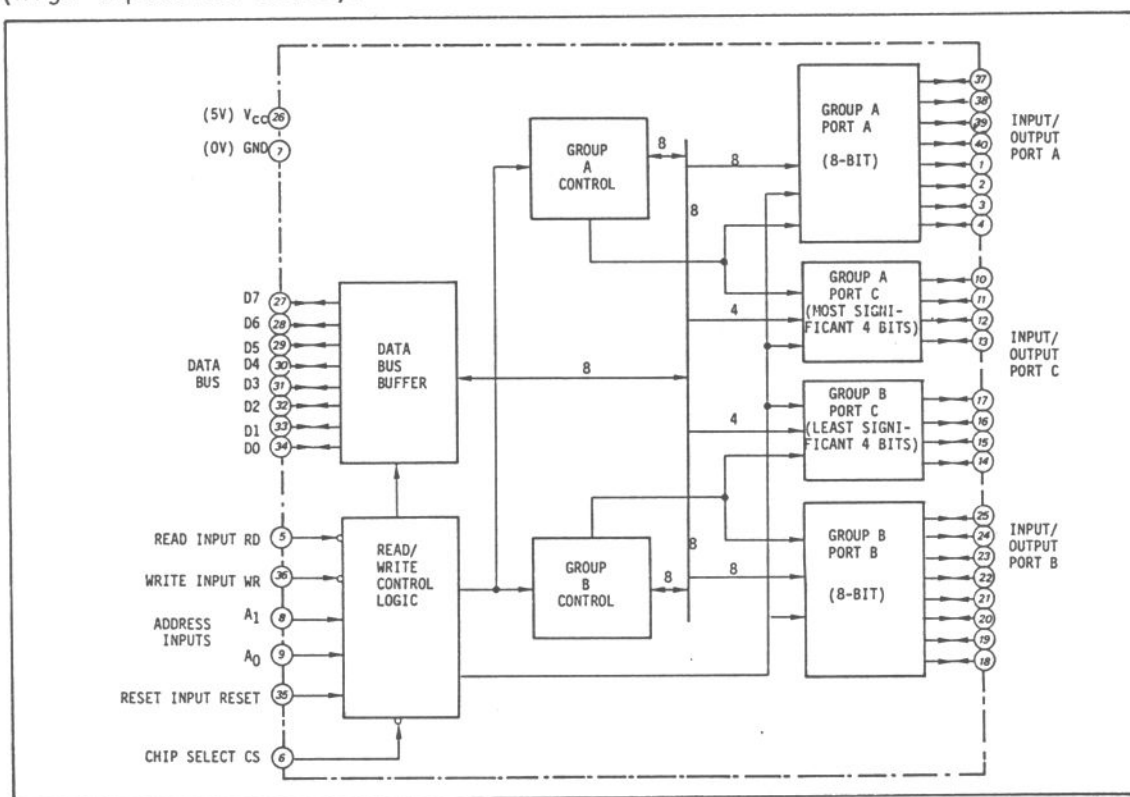
Fig. III-6. Block Diagram

2-5. Functional Description

Data Bus Buffer

This three-state, bidirectional, eight-bit buffer is used to transfer the data when an input or output instruction is executed by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write

The function of this block is to control transfers of both data and control words. It accepts the address signals (A_0, A_1, CS) from the CPU I/O control bus outputs (RD, WR) from the system controller, and RESET signals, and then issues commands to both of the control groups in the PPI.

CS (Chip-Select) Input

At low-level, the communication between the PPI and the CPU is enabled. While at high-level, the data bus is kept in the high-impedance state, so that commands from the CPU are ignored. Then the previous data is kept at the output port.

RD (Read) Input

At low-level, the status or data at the port is transferred to the CPU from the PPI. In essence, it allows the CPU to read data from the PPI.

WR (Write) Input

At low-level, the data or control words are transferred from the CPU and written in the PPI.

A_0, A_1 (Port Address) Input

These input signals are used to select one of the three ports: port A, port B, and port C, or the control register. They are normally connected to the least significant two bits of the address bus.

RESET (Reset) Input

At high-level, all internal registers, including the control register, are cleared. Then all ports are set to the input mode (high-impedance state).

Group A and Group B Control

Accepting commands from the read/write control logic, the control blocks (Group A, Group B) receive 8-bit control words from the internal data bus and issue the proper commands for the associated ports. Control group A is associated with port A and the four high-order bits of port C. Control group B is associated with port B and the four low-order bits of port C. The control register, which stores control words, can only be written into.

Port A, Port B and Port C

The PPI contains three 8-bit ports whose modes and input/output settings are programmed by the system software.

Port A has an output latch/buffer and an input latch. Port B has an I/O latch/buffer and an input buffer. Port C can be divided into two 4-bit ports which can be used as ports for control signals for port A and port B. The basic operations are shown in Table III-2.

A_1	A_0	CS	RD	WR	Operation
0	0	0	0	1	Data bus ← Port A
0	1	0	0	1	Data bus ← Port B
1	0	0	0	1	Data bus ← Port C
0	0	0	1	0	Port A ← Data bus
0	1	0	1	0	Port B ← Data bus
1	0	0	1	0	Port C ← Data bus
1	1	0	1	0	Control register ← Data bus
X	X	1	X	X	Data bus is in high impedance state
1	1	0	0	1	Illegal condition

Table III-2. Basic Operations

Where "0" indicates low level
      "1" indicates high level

Bit Set/Reset

When port C is used as an output port, any one bit of the eight bits can be set (high) or reset (low) by a control word from the CPU. This bit set/reset can be operated in the same way as the mode set, but the control word format is different. This operation is also used for INTE set/reset in mode 1 and 2.

Bit set/reset flag

Active = 0

Don't care

Bit selection code

Port C Bit selected	D_3	D_2	D_1
PC7	1	1	1
PC6	1	1	0
PC5	1	0	1
PC4	1	0	0
PC3	0	1	1
PC2	0	1	0
PC1	0	0	1
PC0	0	0	0

Set/reset code

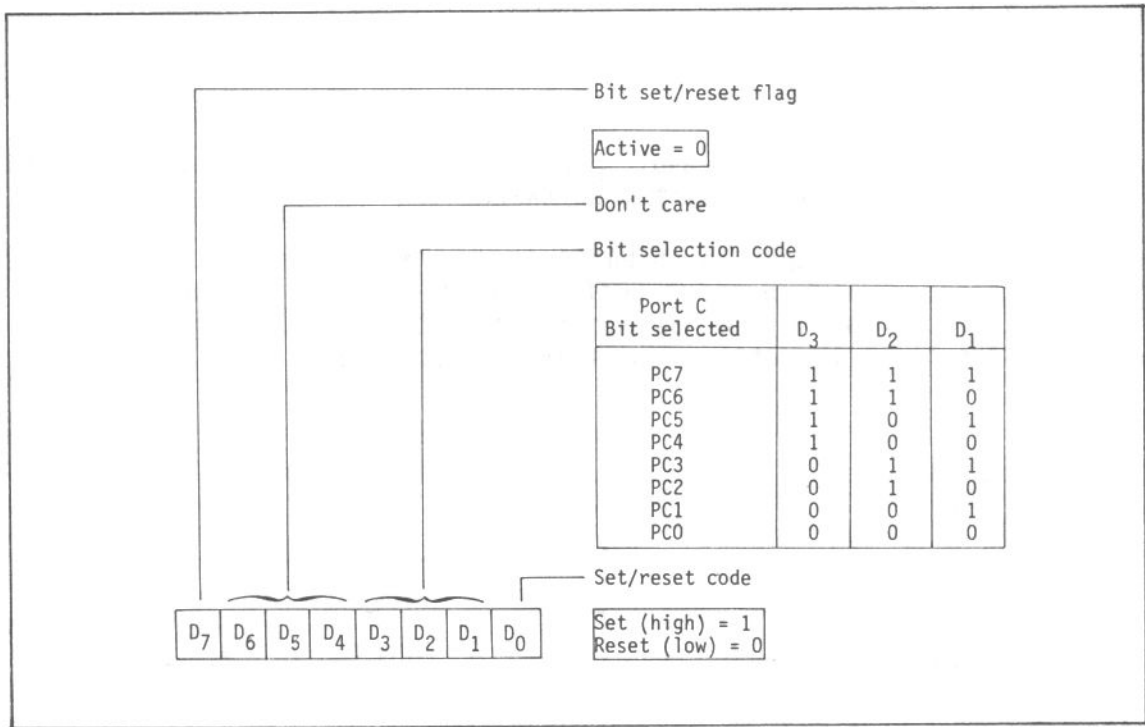| D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

Set (high) = 1
Reset (low) = 0

Fig. III-7. Control Word Format for Port C Set/Reset

3. The TMS 9918 VDP

3-1. Introduction

3-1-1. Description

The TMS9918A/9928A/9929A video display processors (VDP) are N-channel MOS LSI devices used in video systems where data display on a raster-scanned home colour television set or colour monitor is desired. These devices generate all necessary video, control, and synchronization signals and also control the storage, and refresh of display data in the dynamic screen refresh memory. The interfaces to the microprocessor, refresh memory, and the TV require a minimum of additional electronics for the TMS9918A.

In Section 3-4, there is a list of acronyms and a glossary of terms used in this manual.

The TMS9928A/9929A VDPs are functionally identical to the TMS9918A except that the NTSC colour encoding circuitry has been removed and replaced with luminance and colour difference signals. The TMS9918A is pin-for-pin compatible with the TMS9928A/9929A, except for three pins, the composite video output, the external video input and the CPU clock output. These pins are replaced with the Black/White luminance and composite sync (Y) output and two colour difference pins, Blue (B-Y) and Red (R-Y) outputs, respectively. The colour difference outputs allow the user to generate Red-Green-Blue (R-G-B) drive for direct colour gun control, or composite video for use with NTSC or PAL video colour monitor. However, to connect these three outputs to a R-G-B or monitor requires additional R-G-B or encoder circuitry.

The TMS9918A/9928A have a 525-line format for U.S. televisions while the TMS9929A has a 625-line format for use with the European PAL system.

The VDP has four video display modes: Graphics I, Graphics II, Multicolour and Text mode. The text mode provides twenty-four 40-character rows in two colours and is intended to maximize the capacity of the TV screen to display alphanumeric characters. The Multicolour mode provides an unrestricted 64 x 48 colour-dot display employing 15 colours plus transparent. The Graphics II mode is an enhancement of Graphics I mode, allowing it to generate more complex colour and pattern displays. The four video display modes are described in detail in Section 3-2-4.

The video display consists of 35 planes: external VDP, backdrop, pattern plane, and 32 sprite Planes. The planes are vertically stacked with the external VDP being the bottom or innermost plane. The backdrop plane is the next plane followed by the pattern plane that contains Graphics I and Graphics II patterns with the 32 Sprite Planes as the top planes.

The TMS9918A/9928A/9929A VDPs use either a 4K, 8K, or 16K-type low-cost dynamic memory (TMS4027, TMS4108, TMS4116) for storage of the display parameters.

The TMS9918A, TMS9928A, and TMS9929A interface identically to the host microprocessor making them software compatible. Thus, all references to VDP in this document apply to all three devices, except where noted.

3-1-2. Features

- Single-chip solution for interfacing colour TVs (excluding Random-Access Memory (RAM) and Radio Frequency (RF) modulator (TMS9918A only)
- 256 x 192 resolution on TV screen
- 15 unique colours plus transparent
- General 8-bit bidirectional interface to Central Processor Unit (CPU)
- Direct wiring to 4K, 8K or 16K dynamic RAM memories
- Automatic and transparent refresh of dynamic RAMs
- Multiple VDP systems capability
- External VDP input capability (TMS9918A only)
- Composite video output (TMS9918A only)
- Unique planar representation for 3D simulation
- Standard 40-pin package
- Colour difference outputs allow RGB drive - TMS9928A/9929A

3-1-3. Typical Applications

- Colour computer terminals
- Home computers
- Drafting/design aids
- Industrial process monitoring
- Home educational systems
- Animation aids
- European 625-line TV (TMS9929A only)

The following example of a typical application may help introduce the user to the TMS9918A VDP. Figure III-6 is a block diagram of a typical application. Each of the concepts presented in the example is described more fully in later sections of this manual.

The VDP basically has three interfaces: CPU, colour monitor, and dynamic refresh RAM (VRAM), the contents of which define the TV image. The TMS9918A also has eight write-only registers and a read-only status register.

The VDP communicates with the CPU via an 8-bit bidirectional data bus. Three control lines, decoded from the CPU address and enable lines, determine interpretation of the bus. Through the bus, the CPU can write to VRAM, read from VRAM, write to VDP registers, and read the VDP status. The VDP also generates an interrupt signal after every refresh of the TV display.

The dynamic RAM interface consists of direct wiring of eight 4K x 1.8K x 1 or 16K x 1 dynamic RAS/CAS-type RAMs to the VDP. The amount of RAM required is dependent upon the features selected for use in the application.

The interface to the monitor can consist of either wiring the TMS9918A's composite video output pin (suitably buffered) to the input of a colour or black-and-white monitor, or using an appropriate RF modulator to feed the signal into a TV antenna terminal. The TMS9928A/9929A require additional encoder circuitry to interface to a RGB or to a composite video monitor.
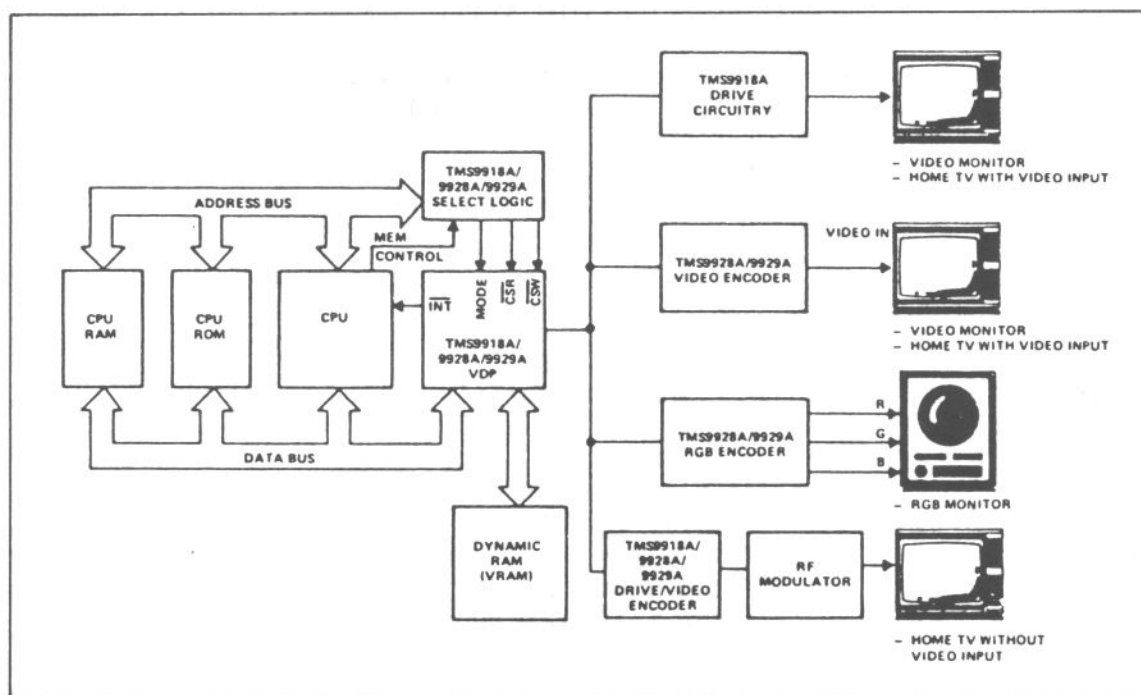
Fig. III-8. System Block Diagram

The VDP operates in four modes and each one can affect the way the VRAM is mapped onto the television screen. In Graphics I and II modes, characters are mapped onto the screen in 8 x 8 pixel blocks yielding 24 lines of 32 blocks (pattern positions) each. In Text mode, there are 24 lines of 40 blocks, each of which is 6 - 8 pixels. In Multicolour mode, there are 48 lines of 64 blocks, each of which is composed of 4 x 4 picture elements (pixels), all of one solid colour. In addition to these, sprites can be superimposed onto the television image in Graphics I, II, and Multicolour mode. Furthermore, signals entering the TMS9918A through the external VDP input can be used as a background to the TMS9918A.

ACRONYMS AND GLOSSARY

B-Y	Blue colour difference output
COMVID	Contains luminance, chrominance and all sync pulse
(Composite Video)	necessary for horizontal and vertical timing
CAS	Column Address Strobe
CPU	Central Processor Unit
CSR	CPU from VDP read select
CSW	CPU to VDP write select
CPULCLK	XTAL - 3
GROMCLK	XTAL - 24

LSB	Least Significant Bit
LSI	Large Scale Integration
MOS	Metal Oxide Semiconductor
MHz	Megahertz
MSB	Most Significant Bit
NTSC	National Television Standards Committee which specifies television signal standards for the USA.
PAL	Phase Alternating Line
Pixel	Picture Element - the smallest point on the TV screen that can be independently controlled.
RAM	Random-Access Memory
RAS	Row-Address Strobe
RASTER	The area in which an image is reproduced
RF	Radio Frequency
R-G-B	Red-Green-Blue
ROM	Read-Only Memory
R/W	Read/Write
R-Y	Red Colour difference Output
Sprite	An object whose pattern is relative to a specified X, Y coordinate and whose position can therefore be controlled by that coordinate with a positional resolution of one pixel.
VDP	Video Display Processor
VRAM	Video RAM, refers to the dynamic RAMs that connect to the VDP and whose contents define the TV image.
Y	Black/white luminance and composite sync.

3-2. Architecture

The TMS9918A video display processor (VDP) is designed to provide a simple interface between a microprocessor and a raster-scanned colour television. The TMS9928A/9929A VDP's are designed as a simple interface between a microprocessor, and R-G-B monitor or video encoder which produces the video monitor. Fig. III-9 is a block diagram of the major portions of the VDP architecture interfaces to the VDP, CPU, VRAM and colour television.

3-2-1. CPU Interface

The VDP interface to the CPU using an 8-bit bidirectional data bus, three control lines, and an interrupt is shown in figure III-10. Through this interface the CPU can conduct four operations:

(1) Write data bytes to VRAM

(2) Read data bytes from VRAM

(3) Write to one of the eight VDP write-only registers

(4) Read the VDP Status Register

Each of these operations requires one or more data transfers to take place over the CPU/VDP data bus interface. The interpretation of the data transfer is determined by the three control lines of the VDP.

NOTE

The CPU can communicate with the VDP simultaneously and asynchronously with the VDP's TV screen refresh operations. The VDP performs memory management and allows periodic intervals of CPU access to VRAM even in the middle of a raster scan.

3-2-1-1. CPU Interface Control Signals

The type and direction of data transfers are controlled by the CSW, CSR, and MODE inputs CSW is the CPU to VDP write select. When it is active (low), the eight bits on CD0 - CD7 are strobed into the VDP. CSR is the CPU from VDP read select. When it is active (low), the VDP outputs eight bits on CD0-CD7 to the CPU. CSW and CSR should never be simultaneously low at the same time. If both are low, the VDP outputs data on CDQ-CD7 and latches in invalid data.

MODE determines the source or destination of a read or write data transfer. MODE is normally tied to a CPU low order address line (A14 for TMS9900).

3-2-1-2. CPU Write to VDP Register

The VDP has eight write-only registers and one read-only status register. The write-only registers control the VDP operation and determine the way in which VRAM is allocated. The status register contains interrupt, sprite coincidence and fifth sprite status flags.

Each of the eight VDP write-only registers can be loaded using two 8-bit data transfers from the CPU. Table III-3 describes the required format for the two bytes. The first byte transferred is the data byte, and the second byte tranferred controls the destination. The MSB of the second byte must be a 1. The next four bits are 0s, and the lowest three bits make up the destination register number. The MODE input is high for both byte transfers.

To rewrite the data in an internal register after a byte of data has already been loaded, the status register must be read so that internal CPU interface logic is reinitialized and will accept the next byte as data and not as a register destination. This situation may be encountered in interrupt-driven program environments. Whenever the status of VDP write parameters is in question, this procedure should be used.

The CPU address is destroyed by writing to the VDP register.

Fig. III-9. VDP Block Diagram
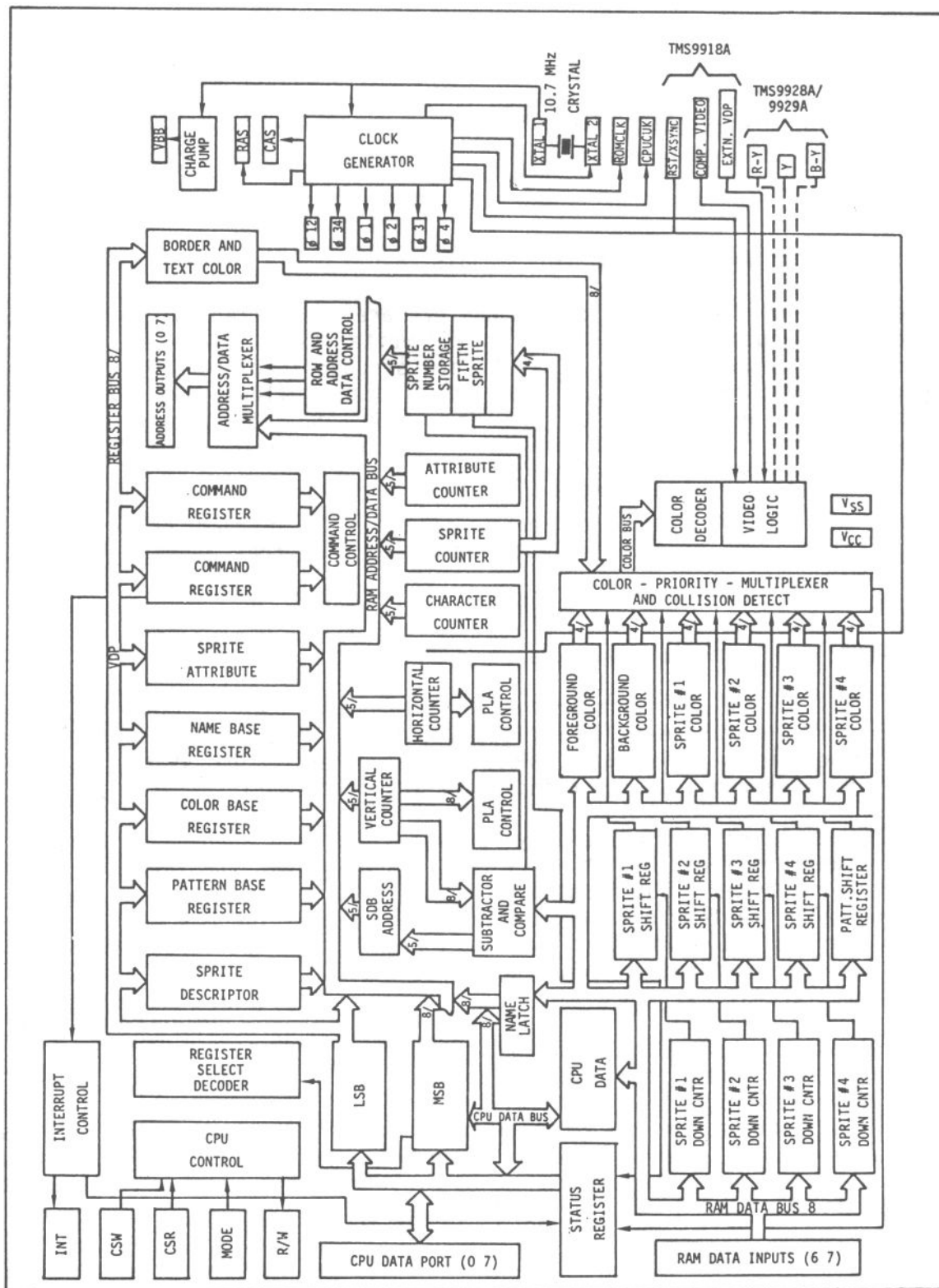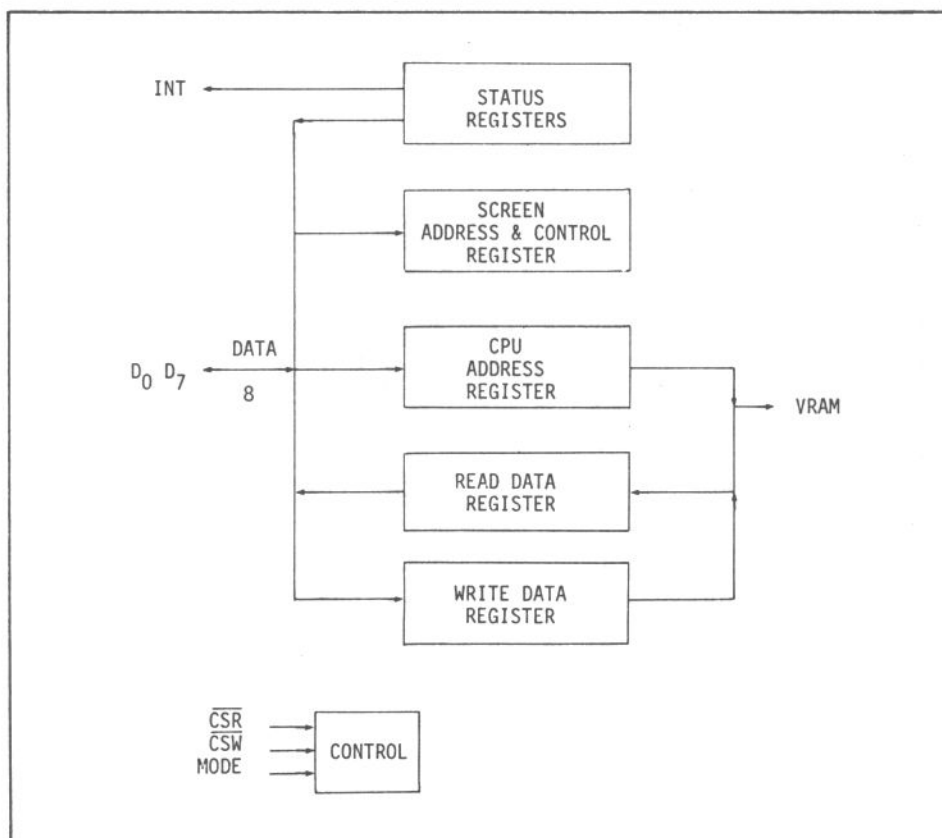
Fig. III-10. VDP to CPU Interface

3-2-1-3. CPU Write to VRAM

The CPU transfers data to the VRAM through the VDP using a 14-bit autoincrementing address register. The address register setup requires 2-byte transfers. A 1-byte transfer is then required to write the data to the addressed VRAM byte. The address register is then autoincremented. Sequential VRAM writes require only 1-byte transfers since the address register is already set up. During setup of the address register, the two MSB's of the second address byte must be 0 and 1 respectively. MODE is high for both address transfers and low for the data transfer. CSW is used in all transfers to strobe the 8 bits into the VDP. See Table III-3.

OPERATION	BIT								CSW	CSR	MODE
	0	1	2	3	4	5	6	7			
WRITE TO VDP REGISTER											
BYTE 1 DATA WRITE	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	0	1	1
BYTE 2 REGISTER SELECT	1	0	0	0	0	RS_0	RS_1	RS_2	0	1	1
WRITE TO VRAM											
BYTE 1 ADDRESS SET UP	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	0	1	1
BYTE 2 ADDRESS SET UP	0	1	A_0	A_1	A_2	A_3	A_4	A_5	0	1	1
BYTE 3 DATA WRITE	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	0	1	0
READ FROM VDP REGISTER											
BYTE 1 DATA READ	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	1	0	1
READ FROM VRAM											
BYTE 1 ADDRESS SET UP	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	0	1	1
BYTE 2 ADDRESS SET UP	0	0	A_0	A_1	A_2	A_3	A_4	A_5	0	1	1
BYTE 3 DATA READ	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	1	0	0

Table III-3. CPU/VDP Data Transfers

3-2-1-4. CPU Read from VDP Status Register

The CPU can read the contents of the status register with a single-byte transfer. MODE is high for the transfer. CSR is used to signal the VDP that a read operation is required.

3-2-1-5. CPU Read from VRAM

The CPU reads from the VRAM through the VDP using the autoincrementing address register. A 1-byte transfer is then required to read the data from the addressed VRAM byte. The address register is then autoincremented. Sequential VRAM data reads require only a 1-byte transfer since the address register is already set up. During setup of the address register, the two MSBs of the second address byte must be 0. By setting up the address this way, a read cycle to VRAM is initiated transfers and low for the data transfers. The VDP requires approximately 8 microseconds to fetch the VRAM byte following the last data transfer and 2 microseconds following address setup.

The CPU interacts with VRAM memory through the VDP. The amount of time necessary for the CPU to transfer a byte of data to or from VRAM memory can vary from 2 to 8 microseconds. Once the VDP has been told to read or write a byte of data to or from VRAM it takes approximately 2 microseconds until the VDP is ready to make the data transfer. In addition to this 2 microsecond delay, the VDP must wait for a CPU address window; i.e., the period of time when the VDP is not occupied with memory refresh or screen display and is available to read or write data.

The worst case time between windows occurs during the Graphics I or Graphics II mode when sprites are being used. During the active display. CPU windows occur once every 16 memory cycles giving a maximum delay of 6 microseconds (a memory cycle takes about 372 nanoseconds). In the Text mode the CPU windows occur at least once out of every three memory cycles or a worst case delay of about 1.1 microseconds. Finally, in the Multicolour mode, CPU windows occur at least once out of every four memory cycles.

If the user needs to access memory in 2 microseconds, two situations occur where the time waiting for an access window is effectively zero. Both of these are independent of the display mode being used.

The first situation occurs when the blank bit of register 1 is 0. With this bit low, the entire screen will show only border colour and the VDP does not have to wait for a CPU access window at any time.

The second situation occurs when the VDP is in the vertical refresh mode. The VDP issues an interrupt output at the end of each active area. This signal indicates that the VDP is entering the vertical refresh mode and that for the next 4.3 milliseconds there is no waiting for an access window. If the user wants the CPU to access memory during this interval, it is necessary for the controlling CPU to monitor the interrupt output of the VDP (the CPU can either poll this output to use it as an interrupt input).

The programme that monitors the interrupt output must allow for its own delays in responding to the interrupt signal and recognize how much time it has left during the 4300 microsecond refresh period. The CPU must write a 1 to the interrupt enabler bit of Register 1 in order to enable the interrupt for each frame, and then read the status register each time an interrupt is issued to clear the interrupt output. A summary of these delay times is presented in Table III-4.

CONDITION	MODE	VDP DELAY	TIME WAITING FOR AN ACCESS WINDOW	TOTAL TIME
Active Display Area	Text	2 µs	0 - 1.1 µs	2 - 3.1 µs
Active Display Area	Graphics I, II	2 µs	0 - 5.95 µs	2 - 8 µs
4300 µs after Vertical Interrupt Signal	All	2 µs	0 µs	2 µs
Register I Blank Bit 0	All	2 µs	0 µs	2 µs
Active Display Area	Multicolour	2 µs	0 - 1.5 µs	2 - 3.5 µs

Table III-4. Memory Access Delay Times

3-2-1-6. VDP-Interrupt

The VDP INT output pin is used to generate an interrupt at the end of each active display scan, which is about every 1/60 second for the TMS9918A/9928A and 1/50 second for the TMS9929A. The INT output is active when the Interrupt Enable bit (IE) in VDP Register 1 is a 1 and

the F bit of the status register is a 1. Interrupts are cleared when the status register is read.

3-2-1-7. VDP Initialization

The VDP is externally intitialized whenever the RESET input is active (low) and must be held low for a minimum of 3 microseconds. The external reset synchronizes all clocks with its falling edge, sets the horizontal and vertical counters to known states, and clears VDP registers 0 and 1. The video display is automatically blanked since the BLANK bit in VDP register 1 becomes a 0. The VDP, however, continues to refresh the VRAM even though the display is blanked. While the RESET line is active, the VDP does not refresh the VRAM.

3-2-2. Write-Only Registers

The eight VDP write-only registers are shown in Fig. III-9. They are loaded by the CPU as described in Section b) Registers 0 and 1 contain flags to enable or disable various VDP features and modes. Registers 2 through 6 contain values that specify starting locations of various sub-blocks of VRAM. The definitions of these sub-blocks are described in section 3-2-4. Register 7 is used to define backdrop and text colours.

Each register is described in the following paragraphs.

3-2-2-1. Register 0

Register 0 contains two VDP option control bits. All other bits are reserved for future use and must be 0s.

BIT 6   M3 (mode bit 3) (see section 3-2-3-2. for table and description)

BIT 7   External VDP enable/disable
                  0 disables external VDP input
                  1 enables external VDP input

NOTE

Enabling bit 7 in the TMS9928A/9929A causes A-Y and B-Y to go to the sync level only when all planes in front of the pixel under question are transparent.

3-2-2-2. Register 1 (contains 8 VDP option control bits)

BIT 0   4/16K selection
                  0 selects 4027 RAM operation
                  1 selects 4108/4116 RAM operation

BIT 1   BLANK enable/disable
                    0 causes the active display area to blank
                    1 enables the active display
                    Blanking causes the display to show border colour only.

BIT 2   IE (Interrupt Enable)
                    0 disables VDP interrupt
                    1 enables VDP interrupt

BIT 3, 4 M1, M2 (mode 1 and 2)
          M1, M2 and M3 determine the operating mode of the VDP:

M1	M2	M3	
0	0	0	Graphics I mode
0	0	1	Graphics II mode
0	1	0	Multicolour mode
1	0	0	Text mode

BIT 5   Reserved

BIT 6   Size (sprite size select)
                    0 selects Size 0 sprites (8 x 8 bit)
                    1 selects Size 1 sprites (16 x 16 bits)

BIT 7   MAG (Magnification option for sprites)
                    0 selects MAG0 sprites (1x)
                    1 selects MAG1 sprites (2x)

REGISTER	MSB							LSB
0	0	0	0	0	0	0	M3	EV
1	4/16K	BLANK	IE	M1	M2	0	SIZE	MAG
2	0	0	0	0	NAME TABLE BASE ADDRESS			
3	COLOUR TABLE BASE ADDRESS							
4	0	0	0	0	0	PATTERN GENERATOR BASE ADDRESS		
5	0	SPRITE ATTRIBUTE TABLE BASE ADDRESS						
6	0	0	0	0	0	SPRITE PATTERN GEN. BASE ADDRESS		
7		TEST COLOUR$_1$			TEST COLOUR$_0$/BACKDROP COL.			
STATUS (READ ONLY)	F	5S	C	FIFTH SPRITE NUMBER				
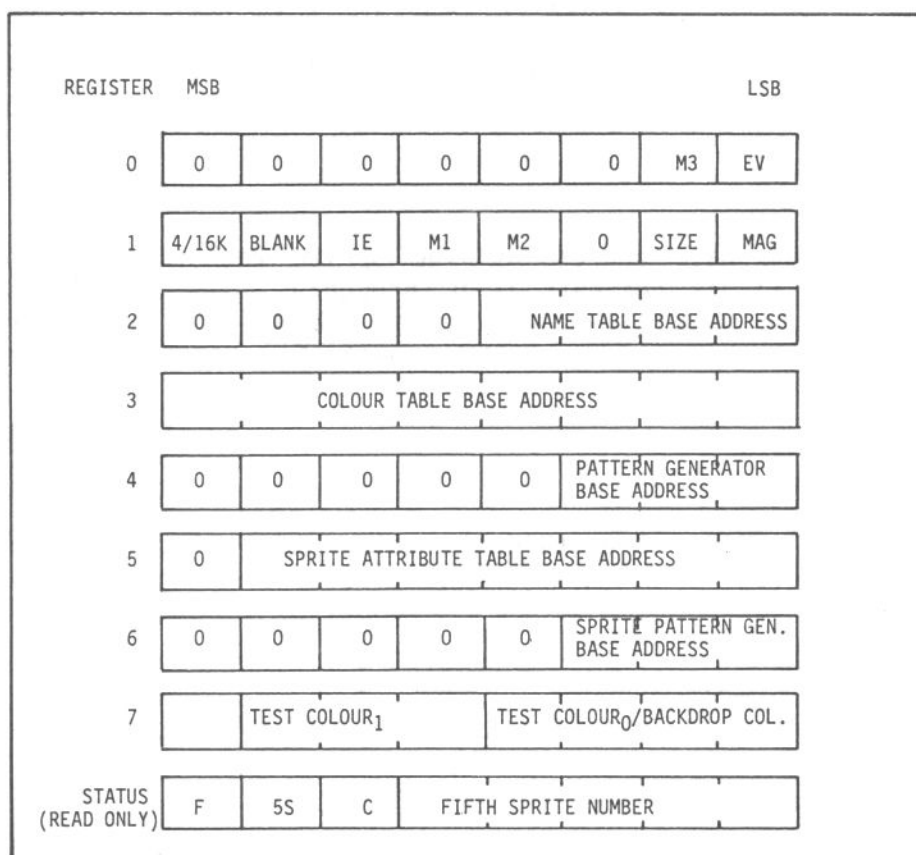
Fig. III-11. VDP Registers

3-2-2-3. Register 2

Register 2 defines the base address of the Name Table sub-block. The range of its contents is from 0 to 15. The contents of the register form the upper 4 bits of the 14-bit Name Table addresses; thus the Name Table base address is equal to (Register 2) *400 (hex).

3-2-2-4. Register 3

Register 3 defines the base address of the Colour Table sub-block. The range of its contents is from 0 to 255. The contents of the register form the upper 89 bits of the 14-bit Colour Table addresses, thus the Colour Table base address is equal to (Register 3) *40 (hex).

3-2-2-5. Register 4

Register 4 defines the base address of the Pattern, text of Multicolour Generator sub-lock. The range of its contents is 0 through 7. The contents of the register form the upper 3 bits of the 14-bit Generator addresses, thus the Generator base address is equal to (Register 4) *800 (hex).

3-2-2-6. Register 5

Register 5 defines the base address of the Sprite Attribute Table sub-lock. The range of its contents is from 0 through 127. The contents of the register form the upper 7 bits of the 14-bit Sprite Attribute Table addresses; thus the base address is equal to (Register 5) *80 (hex).

3-2-2-7-. Register 6

Register 6 defines the base address of the Sprite Pattern Generator sub-lock. The range of its contents is 0 through 7. The contents of the register form the upper 3 bits of the 14-bit Sprite Pattern Generator address; thus the Sprite Pattern Generator base address is equal to (Register 6) *800 (hex).

3-2-2-8. Register 7

The upper 4 bits of Register 7 contain the colour code of colour 1 in the Text mode. The lower 4 bits contain the Colour mode for colour 0 in the Text mode and the backdrop colour in all modes.

3-2-2-9. Setup Values for VDP Registers 2 through 6.

VRAM TABLE ADDRESSING

Register 2 in the VDP contains the starting address for the Name Table sub-block

R2	ADDRESS	
00	0000	
01	0400	
02	0800	
03	0C00	
04	1000	
05	1400	
06	1800	
07	1C00	- Maximum Number for 4K RAMS
08	2000	
09	2400	
0A	2800	
0B	2C00	
0C	3000	
0D	3400	
0E	3800	
0F	3C00	- Maximum Number

Table III-5. R2 * $400_{(16)}$ = START ADDRESS

Register 3 in the VDP contains the starting address for the Colour Table

R3	START ADDRESS	R3	START ADDRESS	R3	START ADDRESS
00	0000	28	0A00	50	1400
01	0040	29	0A40	51	1440
02	0080	2A	0A80	52	1480
03	00C0	2B	0AC0	53	14C0
04	0100	2C	0B00	54	1500
05	0140	2D	0B40	55	1540
06	0180	2E	0B80	56	1580
07	01C0	2F	0BC0	57	15C0
08	0200	30	0C00	58	1600
09	0240	31	0C40	59	1640
0A	0280	32	0C80	5A	1680
0B	02C0	33	0CC0	5B	16C0
0C	0300	34	0D00	5C	1700
0D	0340	35	0D40	5D	1740
0E	0380	36	0D80	5E	1780
0F	03C0	37	0DC0	5F	17C0
10	0400	38	0D00	60	1800
11	0440	39	0E40	61	1840
12	0480	3A	0E80	62	1880
13	04C0	3B	0EC0	63	18C0
14	0500	3C	0F00	64	1900
15	0540	3D	0F40	65	1940
16	0580	3E	0F80	66	1980
17	05C0	3F	0FC0*	67	19C0
18	0600	40	1000	68	1A00
19	0640	41	1040	69	1A40
1A	0680	42	1080	6A	1A80
1B	06C0	43	10C0	6B	1AC0
1C	0700	44	1100	6C	1B00
1D	0740	45	1140	6D	1B40
1E	0780	46	1180	6E	1B80
1F	07C0	47	11C0	6F	1BC0
20	0800	48	1200	70	1C00
21	0840	49	1240	71	1C40
22	0880	4A	1280	72	1C80
23	08C0	4B	12C0	73	1CC0
24	0900	4C	1300	74	1D00
25	0940	4D	1340	75	1D40
26	0980	4E	1380	76	1D80
27	09C0	4F	13C0	77	1DC0

* Maximum number for 4K RAMS

Table III-6. (R3) * $40_{(16)}$ Starting Address

R3	START ADDRESS	R3	START ADDRESS	R3	START ADDRESS
78	1E00	A6	2980	D3	34C0
79	1E40	A7	29C0	D4	3500
7A	1E80	A8	2A00	D5	3540
7B	1EC0	A9	2A40	D6	3580
7C	1F00	A4	2A80	D7	35C0
7D	1F40	AB	2AC0	D8	3600
7E	1F80	AC	2B00	D9	3640
7F	1FC0	AD	2B40	DA	3680
80	2000	AE	2B80	DB	36C0
81	2040	AF	2BC0	DC	3700
82	2080	B0	2C00	DD	3740
83	20C0	B1	2C40	DE	3780
84	2100	B2	2C80	DF	37C0
85	2140	B3	2CC0	E0	2800
86	2180	B4	2D00	E1	3840
87	21C0	B5	2D40	E2	3880
88	2200	B6	2D80	E3	38C0
89	2240	B7	2DC0	E4	3900
8A	2280	B8	2E00	E5	3940
8B	22C0	B9	2E40	E6	3980
8C	2300	BA	2E80	E7	39C0
8D	2340	BB	2EC0	E8	3A00
8E	2380	BC	2F00	E9	1A40
8F	23C0	BD	2F40	EA	3A80
90	2400	BE	2F80	EB	3AC0
91	2440	BF	2FC0	EC	3B00
92	2480	C0	3000	ED	3B40
93	24C0	C1	3040	EE	3B80
94	2500	C2	3080	EF	3BCD
95	2540	C3	30C0	F0	3C00
96	2580	C4	3100	F1	3C40
97	25C0	C5	3140	F2	3C80
98	2600	C6	3180	F3	3CC0
99	2640	C7	31C0	F4	2D00
9A	2680	C8	3200	F5	3D40
9B	26C0	C9	3240	F6	3D80
9C	2700	CA	3280	F7	3DC0
9D	2740	CB	32C0	F8	3E00
9E	2780	CC	3300	F9	3240
9F	27C0	CD	3340	FA	3E80
A0	2800	CE	3380	FB	3EC0
A1	2840	CF	33C0	FC	3F00
A2	2880	D0	3400	FD	3F40
A3	28C0	D1	3440	FE	3F80
A4	2900	D2	3480	FF	3FC0
A5	2940				

Table III-7. (R3)* $40_{(16)}$ Starting Address (Concluded)

Register 4 in the VDP contains the starting address for the Pattern Generator Sub-block.

R4	START ADDRESS	
00	0000	
01	0800	- Max # for 4K RAMS
02	1000	
03	1800	
04	2000	
05	2800	
06	3000	
07	3800	- Max # for 16K RAMS

Table III-8. (R4) * $800_{(16)}$ = Start Address

Register 5 in the VDP contains the starting address for the Sprite Attribute Table.

R5	START ADDRESS	R5	START ADDRESS	R5	START ADDRESS	R5	START ADDRESS
00	0000	21	1080	40	2000	60	3000
01	0080	22	1100	41	2080	61	3080
02	0400	23	1180	42	2100	62	3100
03	0180	24	1200	43	2180	63	3180
04	0200	25	1280	44	2200	64	3200
05	0280	26	1300	45	2280	65	3280
06	0300	27	1380	46	2300	66	3300
07	0380	28	1400	47	2380	67	3380
08	0400	29	1480	48	2400	68	3400
09	0480	2A	1500	49	2480	69	3480
0A	0500	2B	1580	4A	2500	6A	3500
0B	0580	2C	1600	4B	2580	6B	3580
0C	0600	2D	1680	4C	2600	6C	3600
0D	0680	2E	1700	4D	2680	6D	3680
0E	0700	2F	1780	4E	2700	6E	3700
0F	0780	30	1800	4F	2780	6F	3780
10	0800	31	1880	50	2800	70	3800
11	0880	32	1900	51	2880	71	3880
12	0900	33	1980	52	2900	72	3900
13	0980	34	1A00	53	2980	73	3980
14	0A00	35	1A80	54	2A00	74	3A00
15	0A80	36	1B00	55	2A80	75	3A80
16	0B00	37	1B80	56	2B00	76	3B00
17	0B80	38	1C00	57	2B80	77	3B80
18	0C00	39	1C80	58	2C00	78	3C00
19	0C80	3A	1D00	59	2C80	79	3C80
1A	0D00	3B	1D80	5A	2D00	7A	3D00
1B	0D80	3C	1E00	5B	2D80	7B	3D80
1C	0E00	3D	1E80	5C	2E00	7C	3E00
1D	0E80	3E	1F00	5D	2E80	7D	3E80
1E	0F00	3F	1F80	5E	2F00	7E	3F00
1F	0F80*			5F	2F80	7F	3F80
20	1000						

* Maximum number for 4K RAMS

Table III-9. (R5) * $80_{(16)}$ = Start Address

Register 6 contains the value for the starting address of the Sprite Pattern Generator sub-block.

```
┌─────────────────────────────────────────────────────┐
│                                                     │
│   R6    │   START ADDRESS                           │
│  ───────┼──────────────────────                     │
│   00    │   0000                                    │
│   01    │   0800 - Max # for 4K DRAMS               │
│   02    │   1000                                    │
│   03    │   1800                                    │
│   04    │   2000                                    │
│ · 05    │   2800                                    │
│   06    │   3000                                    │
│   07    │   3800 - Max # for 16K RAMS               │
│                                                     │
└─────────────────────────────────────────────────────┘
```

Table III-10. STARTING ADDRESS + R6 * 800

3-2-3. Status Register

The VDP has a single 8-bit status register that can be accessed by the CPU. The status register contains the interrupt pending flag, the sprite coincidence flag, the fifth sprite flag and the fifth sprite number, if one exists. The format of the status register is shown in Figure III-11 and is discussed in the following paragraphs.

The status register may be read at any time to test the F, C, and 5S status bits. Reading the status register will clear the interrupt flag F. However, asychronous reads will cause the frame flag (F) bit to be reset and therefore missed. Consequently, the status register should be read only when the VDP interrupt is pending.

3-2-3-1. Interrupt Flag (F)

The F status flag in the status register is set to 1 at the end of the raster scan of the last line of the active display. It is reset to a 0 after the status register is read or when the VDP is externally reset. If the Interrupt Enable bit in VDP Register 1 is active (1), the VDP interrupt output (INT) will be active (low) whenever the F status is a 1.

Note that the status register needs to be read frame by frame in order to clear the interrupt and receive the new interrupt of the next frame.

3-2-3-2. Coincidence Flag (C)

The C status flag in the status register is set to a 1 if two or more sprites coincide. Coincidence occurs if any two sprites on the screen have one overlapping pixel. Transparent coloured sprites, as well as those that are partially or completely off the screen, are also considered. Sprites beyond the Sprite Attribute Table terminator (D016) are not considered.

The C flag is cleared to a 0 after the status register is read or the VDP is externally reset. The status should be read immediately upon power up to ensure that the coincidence flag is reset.

The VDP checks each pixel position for coincidence during the generation of the pixel regardless of where it is located on the screen. This occurs every 1/60th of a second for the TMS9918A and TMS9928A and every 1/50th of a second for the TMS9929A. Thus, when moving sprites more than one pixel position during these intervals, it is possible for the sprites to have multiple pixels overlapping or even to have passed completely over one another when the VDP checks for coincidence.

3-2-3-3. Fifth Sprite Flag (5S) and Number

The 5S status flag in the status register is set to a 1 whenever there are five or more sprites on a horizontal line (lines 0 to 192) and the frame flag is equal to a 0. The 5S status flag is cleared to a 0 after the status register when the 5S flag is set and is valid whenever the 5S flag is one. The setting of the fifth sprite flag will not cause an interrupt.

3-2-4. Video Display Modes

The VDP displays an image on the screen that can best be envisioned as a set of display planes sandwiched together. Figure III-12 shows the definition of each of the planes. Objects on planes closest to the viewer have higher priority. In cases where two entities on two different planes are occupying the same spot on the screen, the entity on the higher priority plane will show at that point. For an entity on 2 specific plane to show through, all planes in front of that plane must be outside of the sprite itself, are transparent. Since the coordinates of the sprite are in terms of pixels, the sprite can be positioned and moved about very accurately. Sprites are available in three sizes 8 x 8 pixels, 16 x 16 pixels, and 32 x 32 pixels.

Behind the Sprite Planes is the Pattern Plane. The Pattern Plane is used for textual and graphics images generated by the Text, Graphics I, Graphics II, or Multicolour modes. Behind the Pattern Plane is the backdrop, which is larger in area than the other planes so that it forms a border around the other planes. The last and lowest priority plane is the External VDP Plane. Its image is defined by the external VDP input pin which allows the TMS9918A to mix the external video signal internal to the chip.

The mixing must occur outside of the chip for the TMS9928A and TMS9929A. This is achieved through the colour difference outputs swinging to a special level (sync level is shown in fig. III-14) not used by the colour difference signals in normal operation. This occurs when bit 7 of Register 0 is set high. External mixing circuitry is required to detect this change in the level of the colour difference signals and then switch from the VDP signals to an external source's signals (see figure III-15 and III-16).
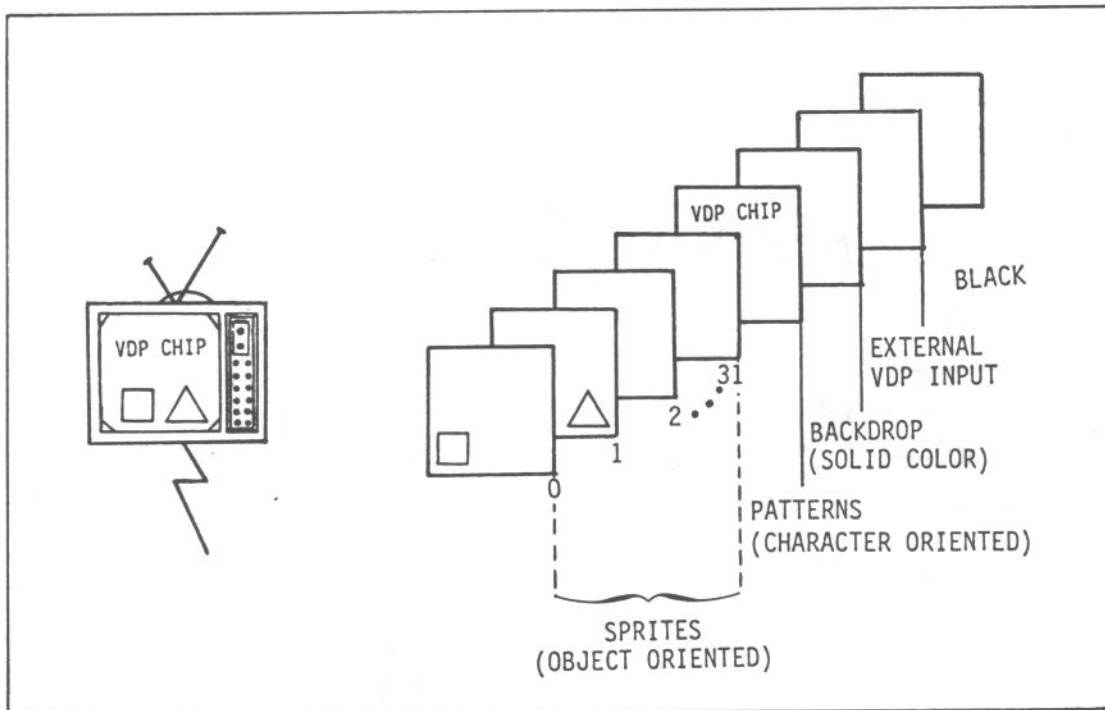
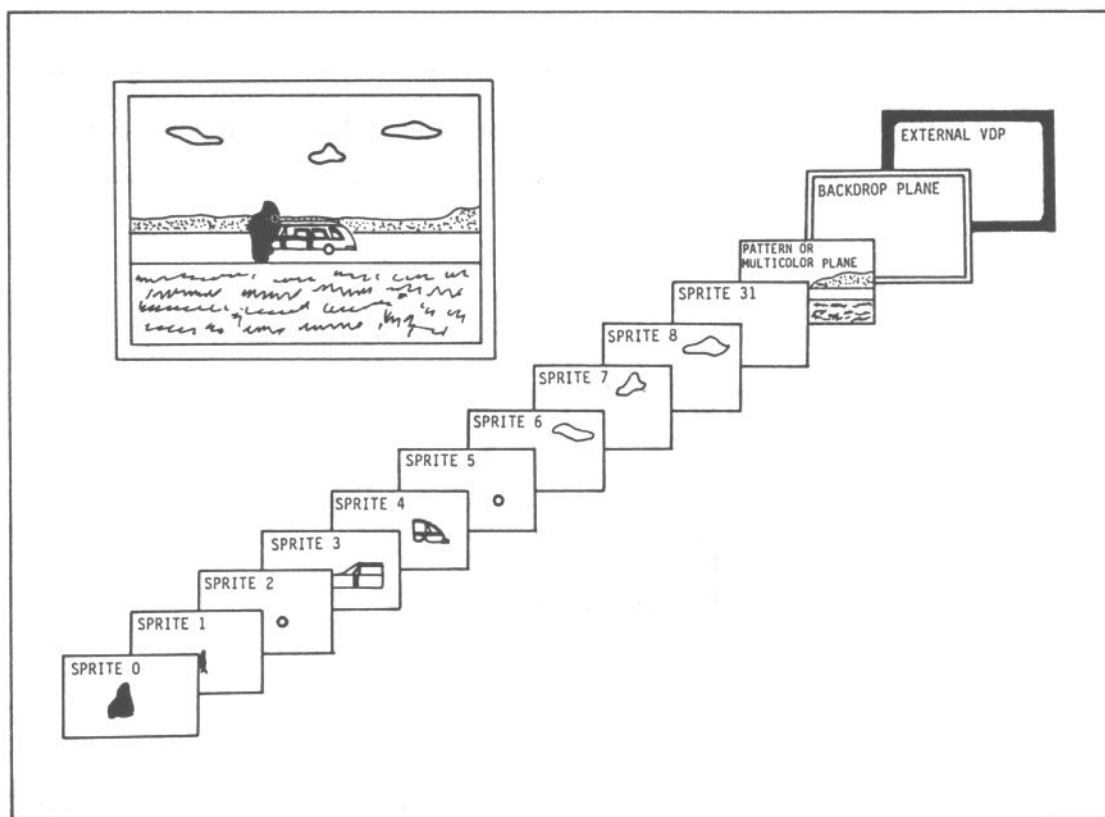Fig. III-12. VDP Display Planes (Definition)

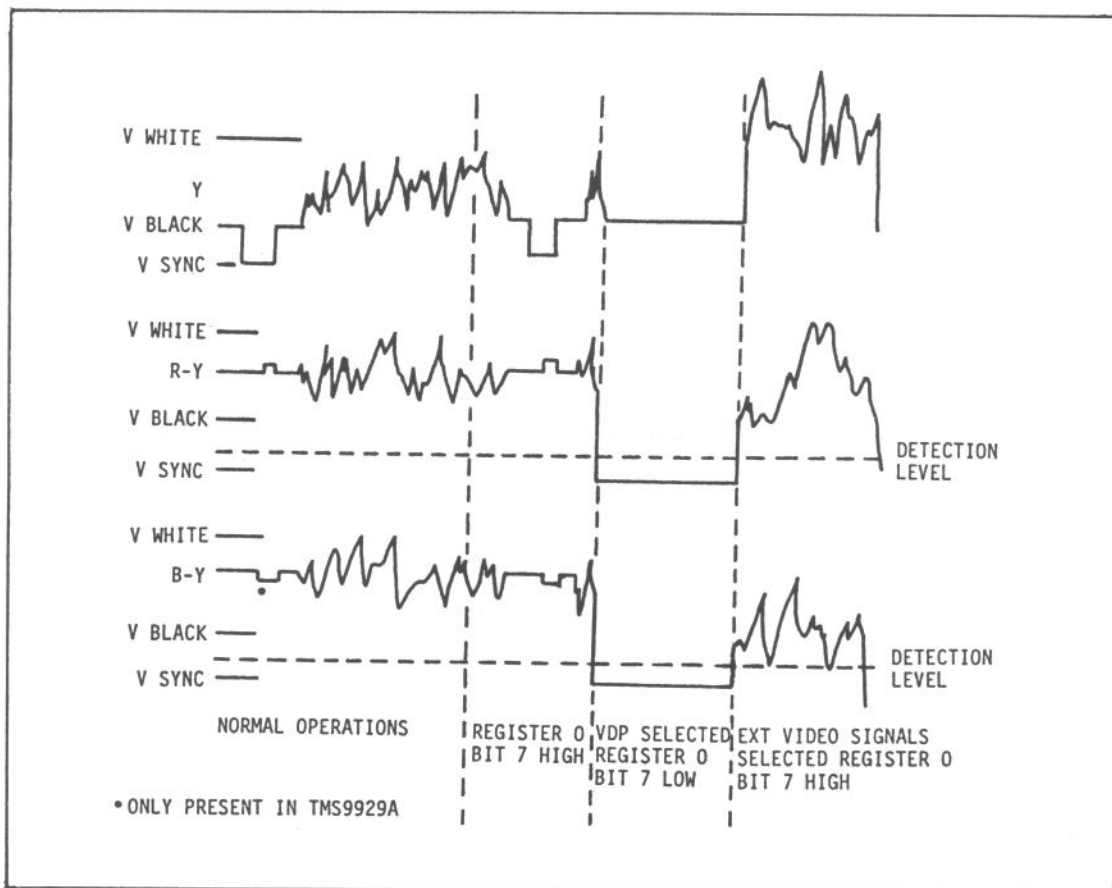Fig. III-13. VDP Display Planes (First 32 Planes)

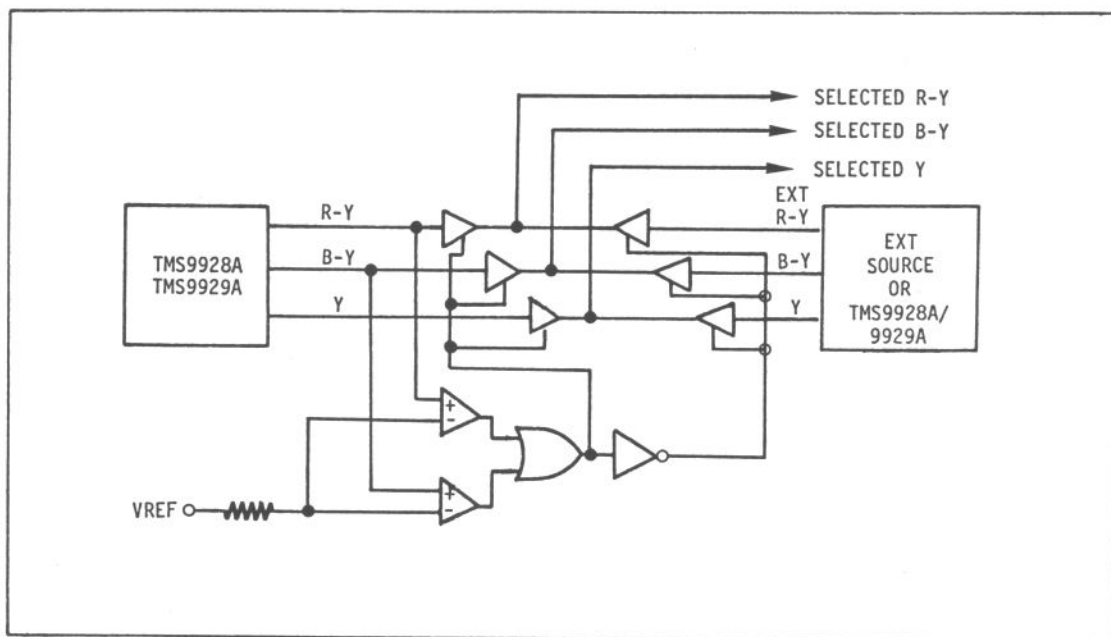Fig. III-14. TMS9928/9929A Signal Waveforms for Multiple VDP Operation

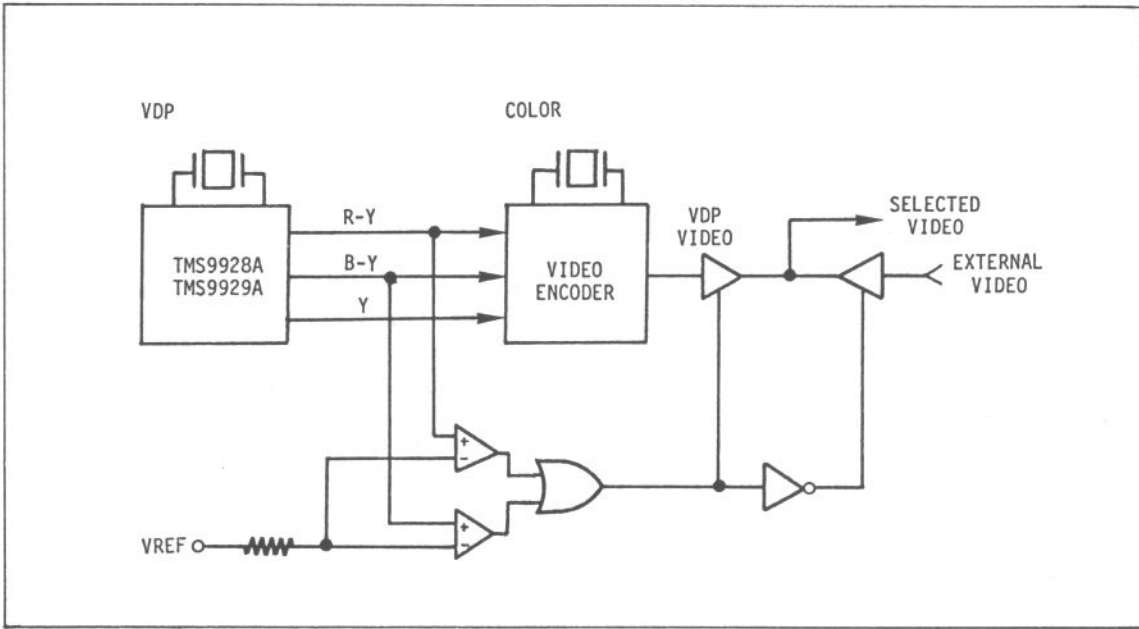Fig III-15. Using Colour Difference Signals to Mix External Colour Difference Type Source

Fig. III-16. Using Colour Difference Signals to Mix External Video Sources

4. The 8910 PSG

4-1. Introduction

It is apparent that any microprocessor is capable of producing acceptable sounds with only a transducer if the processor has no other tasks to perform while the sound is sustained. In real world microprocessor use, however, video games need refreshing, keyboards need scanning, etc. For example, in order to produce a single channel of ninth octave C (8372 Hz) the signal needs attention every sixty microseconds. Software required to produce this simple effect and still perform other activities would in the least be very complex if not impossible. In the extreme, random noise requires periodic attention even more frequently.

This need for software-produced sounds without the constant attention of the processor is now satisfied with the availability of the General Instrument AY-3-8910 and AY-3-8912 Programmable Sound Generators.

4-1-1. Description

The AY-3-8910/8912 Programmable Sound Generator (PSG) is a Large Scale Integrated Circuit which can produce a wide variety of complex sounds under software control. The AY-3-8910/8912 is manufactured in GI's N-channel Ion Implant Process. Operation requires a single 5V power supply, a TTL compatible clock, and a microprocessor controller such as the GI-16-bit CP1600/1610 or one of GI's PIC1650 series of 8-bit microcomputers.

The PSG is easily interfaced to any bus oriented system. Its flexibility makes it useful in applications such as music synthesis, sound effects generation, audible alarms, tone signalling and FSK modems. The analog sound outputs can each provide 4 bits of logarithmic digital to analog conversion, greatly enhancing the dynamic range of the sounds produced.
In order to perform sound effects while allowing the processor to continue its other tasks, the PSG can continue to produce sound after the initial commands have been given by the control processor. The fact that realistic sound production often involves more than one effect is satisfied by the three independently controllable channels available in the PSG.

All of the circuit control signals are digital in nature and intended to be provided directly by a microprocessor/microcomputer. This means that one PSG can produce the full range of required sounds with no change in external circuitry. Since the frequency response of the PSG ranges from sub-audible at its lowest frequency to post-audible at its highest frequency, there are few sounds which are beyond reproduction with only the simplest electrical connections.

Since most applications of a microprocessor/PSG system would also require interfacing between the outside world and the microprocessor, this facility has been designed into the PSG. The AY-3-8910 has two general purpose 8-bit I/O ports and is supplied in a 409 lead package; in the AY-3-8912 has one port and 28 leads.

4-1-2. Features

- Full software control of sound generation.
- Interface to most 8-bit and 16-bit microprocessors
- Three independently programmed analog outputs.
- Two 8-bit general purpose I/O ports (AY-3-8910)
- One 8-bit general purpose I/O port (AZ-3-8912)
- Single +5 Volt Supply.

4-1-3. Scope

This Data Manual is intended to introduce the techniques needed to cause the AY-3-8910/8912 Programmable Sound Generator to perform in its intended fashion. All of the programs, programming, and hardware designs have been tested to ensure that the methods are practical rather than purely theoretical.

Although the techniques described will produce powerful results, the range of sounds to be synthesized is so vast and the PSG capabilities so varied that this guide should be viewed merely as an introduction to the applications possiblities of the PSG.
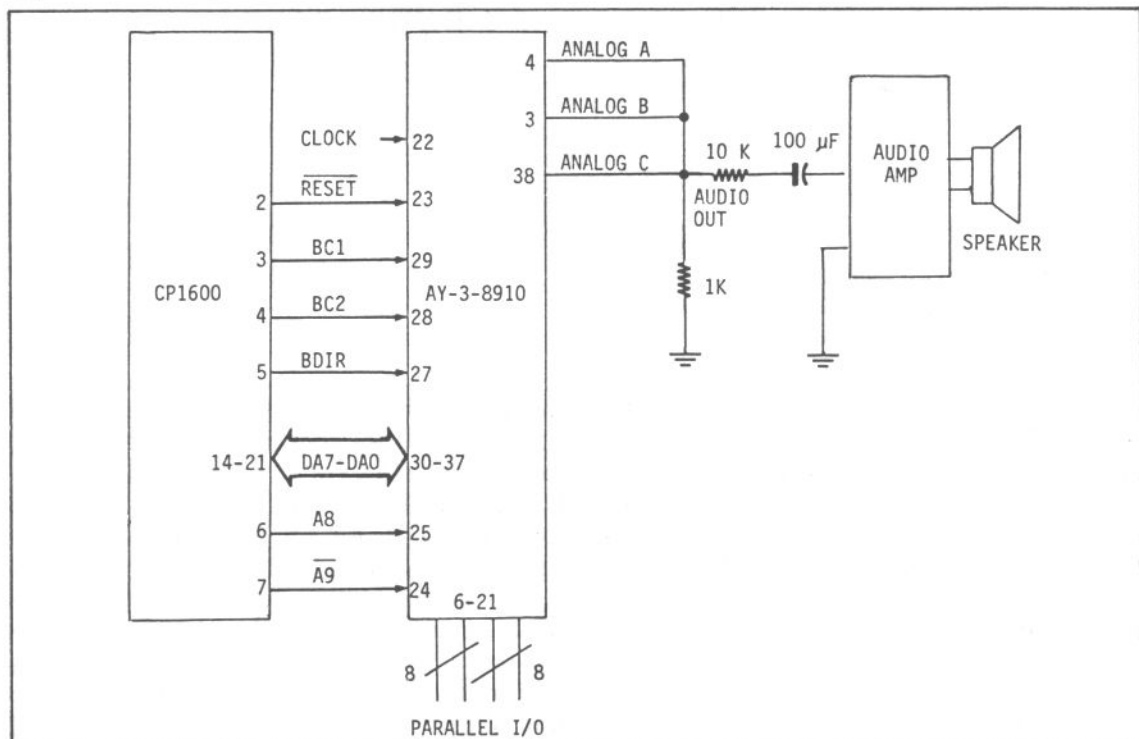

Fig. III-17. Typical System Diagram

4-2. Architecture

The AY-3-8910/8912 is a register oriented Programmable Sound Generator (PSG). Communication between the processor and the PSG is based on the concept of memory-mapped I/O. Control commands are issued to the PSG by writing to 16 memory-mapped register. Each of the 16 registers within the PSG is also readable so that the microprocessor can determine as necessary present states or stored data values.

All functions of the PSG are controlled through its 16 registers which once programmed, generate and sustain the sounds, thus freeing the system processor for other tasks.

4-2-1. Basic Functional Blocks

The principal element of the PSG is the array of 16 read/write control registers. These 16 registers look to the CPU as a block of memory and as such occupy a 16 word block out of 1.024 possible addresses. The 10 address bits (8 bits on the common data/address bus and 2 separate address bits A8 and A9) are decoded as follows

*$\overline{A9}$	A8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
0	1	0	0	0	0	0	0	0	0

*$\overline{A9}$ is not provided on the AY-3-8912

THROUGH

0	1	0	0	0	0	1	1	1	1

HIGH ORDER (Chip Select)   LOW ORDER (Register)
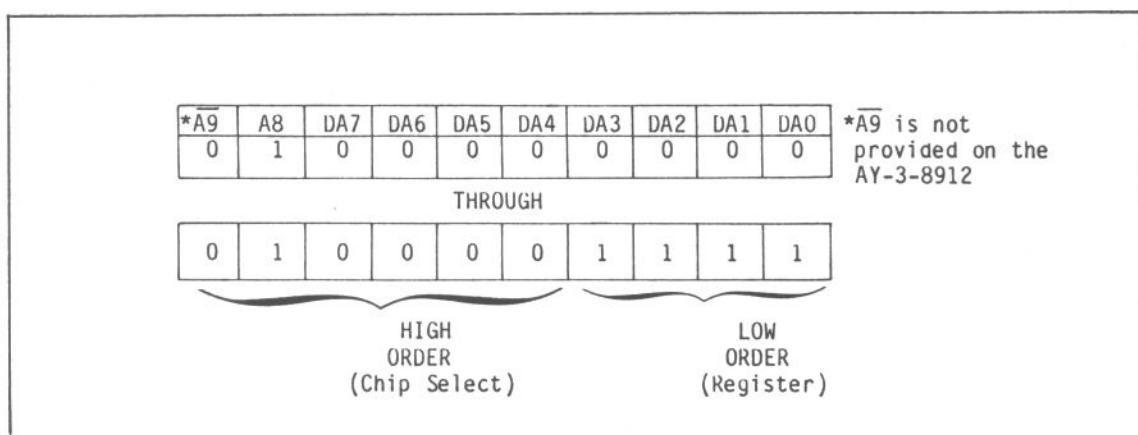
Fig. III-18.

The four low order address bits select one of the 16 registers (R0--R17$_8$). The six high order address bits function as "chip selects" to control the tri-state bidirectional buffers (when the high order address bits are "incorrect", the bidirectional buffers are forced to a high impedance state). High order address bits A9 A8 are fixed in the PSG design to recognize a 01 code; high order address bits DA7--DA4 may be mask-programmed to any 4-bit code by a special order factory maks modification. Unless otherwise specified, address bits DA7-DA4 are programmed to recognize only a 0000 code. A valid high order address latches the register address (the low order 4 bits) in the Register Address Latch/Decoder block. A latched address will remain valid until the receipt of a new address, enabling multiple reads and writes of the same register contents without the need for redundant re-addressing.

A9 A8 BDIR BC2 BC1 DA7 DA0 RESET

RESET
REGISTERS

8

8

BUS
CONTROL
DECODE

BI-DIRECTIONAL
BUFFERS

REGISTER	BIT	B7	B6	B5	B4	B3	B2	B1	B0
R0	Channel A Tone Period	8-BIT Fine Tune A							
R1						4-BIT Coarse Tune A			
R2	Channel B Tone Period	8-BIT Fine Tune B							
R3						4-BIT Coarse Tune B			
R4	Channel C Tone Period	8-BIT Fine Tune C							
R5						4-BIT Coarse Tune C			
R6	Noise Period				5-BIT Period Control				
R7	Enable	IOB IN	IOA OUT	Noise C	Noise B	Noise A	Tone C	Tone B	Tone A
R10	Channel A Amplitude				M	L3	L2	L1	L0
R11	Channel B Amplitude				M	L3	L2	L1	L0
R12	Channel C Amplitude				M	L3	L2	L1	L0
R13	Envelope Period	8-BIT Fine Tune E							
R14		8-BIT Coarse Tune E							
R15	Envelope Shape/Cycle					CONT	ATT	ALT	HOLD
R16	I/O Port A Data Store	8-BIT PARALLEL I/O on Port A							
R17	I/O Port B Data Store	8-BIT PARALLEL I/O Port B							

REGISTER
ADDRESS
LATCH/
DECODER

REGISTER ARRAY
(16 READ WRITE
CONTROL REGISTERS)

8   8   2

I/O PORT B   I/O PORT A

8   8
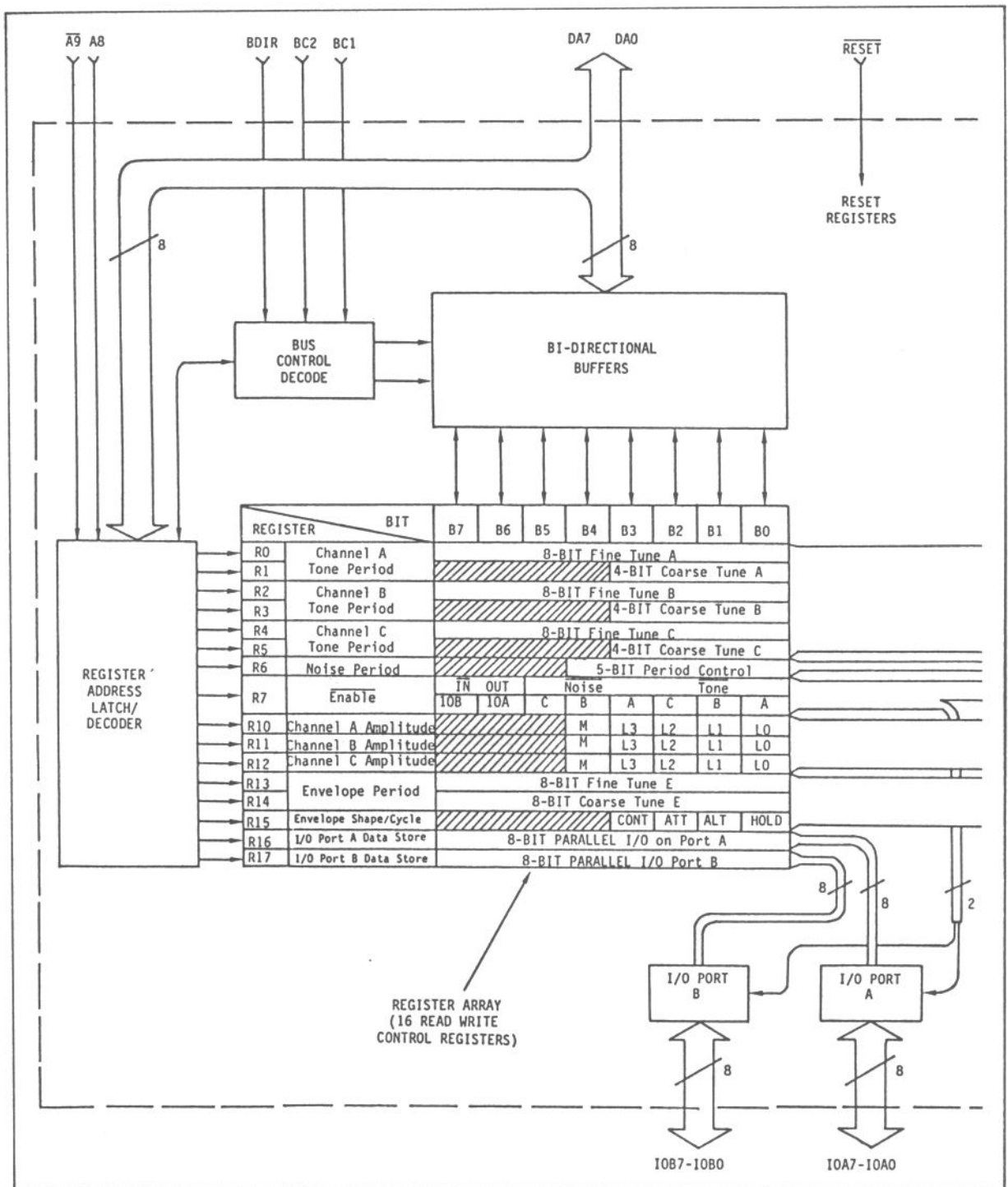
IOB7-IOB0   IOA7-IOA0

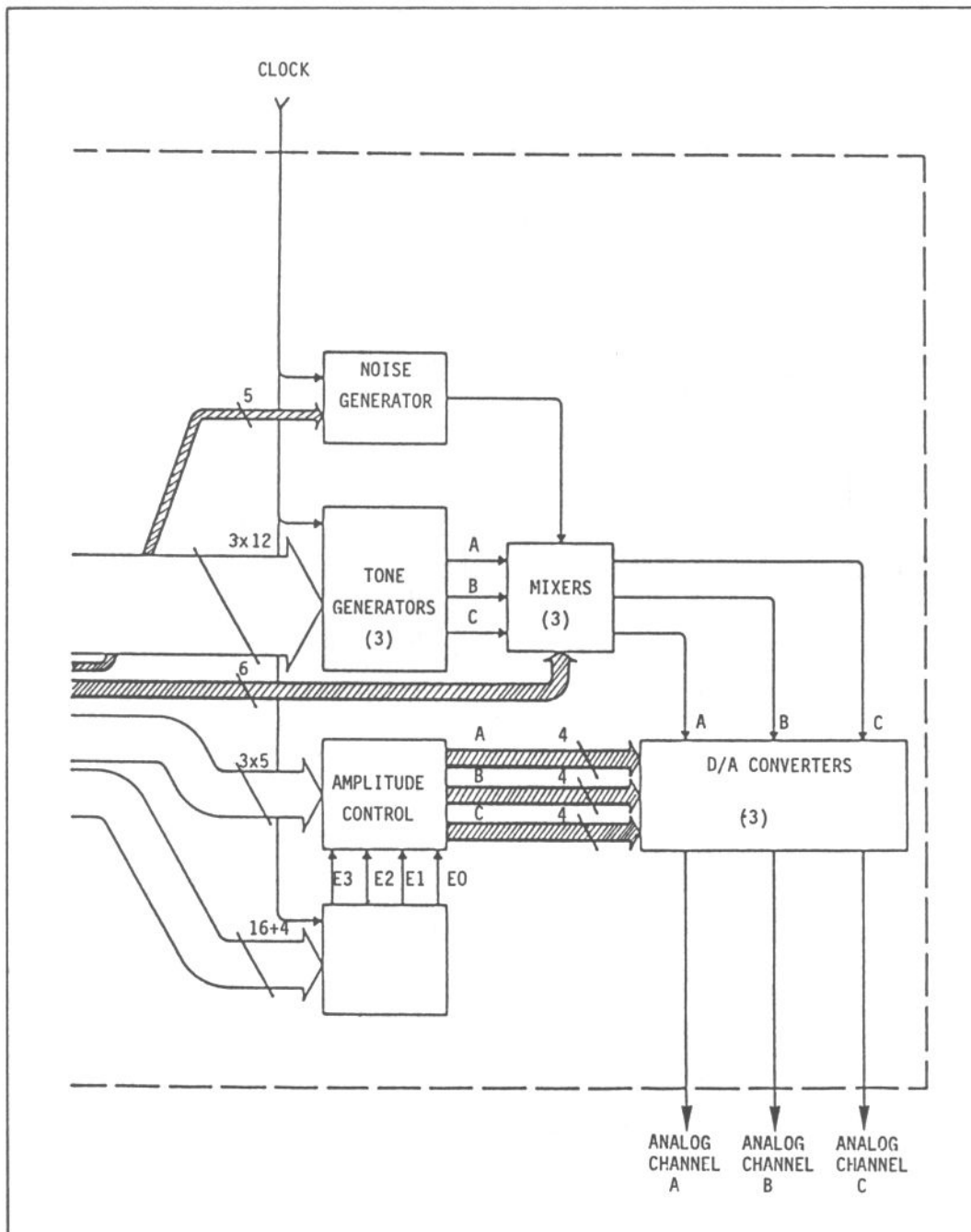Fig. III-19 a. PSG Block Diagram

- 71 -

Fig. III-19 b.

Conditioning of the Register Address Latch/Decoder and the Bidirectional Buffers to recognize the bus function required (inactive, latch address, write data, or read data) is accomplished by the Bus Control Decode block.

The function of each of the 16 PSG registers and the data flow of each register°s contents are shown in contents in Fig. III-17 and explained in detail in Section 4-3, "Operation". For reference purposes, the Register Array details are reproduced in Fig. III-19 a & b.

4-2-1-2. Sound Generating Blocks

The basic blocks in the PSG which produce the programmed sounds include:

Tone Generators   produce the basic square wave tone frequencies for each channel (A, B, C)

Noise Generator   produces a frequency modulated pseudo random pulse width square wave output.

Mixers   combine the outputs of the Tone Generators and the Noise Generator. One for each channel (A, B, C).

Amplitude Control   provides the D/A Converters with either a fixed or variable amplitude pattern. The fixed amplitude is under direct CPU control; the variable amplitude is accomplished by using the output of the Envelope Generator.

Envelope Generator   produces an envelope pattern which can be used to amplitude modulate the output of each Mixer.

D/A Converters   the three D/A Converters each produce up to a 16 level output signal as determined by the Amplitude Control.

4-2-1-3. I/O Ports

Two additional blocks are shown in the PSG Block Diagram which have nothing directly to do with the production of sound - these are the two I/O Ports (A and B). Since virtually all uses of microprocessor-based sound would require interfacing between the outside world and the processor, this facility has been included in the PSG. Data to/from the CPU bus may be read/written to either of two 8-bit I/O Ports without affecting any other function of the PSG. The I/O Ports are TTL-compatible and are provided with internal pull-ups on each pin. Both Ports are available on the AY-3-8910; only I/O Port A is available on the AY-3-8912.

REGISTER	BIT	B7	B6	B5	B4	B3	B2	B1	B0
R0	Channel A	8-BIT Fine Tune A							
R1	Tone Period	/////	/////	/////	/////	4-BIT Coarse Tune A			
R2	Channel B	8-BIT Fine Tune B							
R3	Tone Period	/////	/////	/////	/////	4-BIT Coarse Tune B			
R4	Channel C	8-BIT Fine Tune C							
R5	Tone Period	/////	/////	/////	/////	4-BIT Coarse Tune C			
R6	Noise Period	/////	/////	/////	5-BIT Period Control				
R7	$\overline{\text{Enable}}$	IN 10B	OUT 10A	Noise C	Noise B	Noise A	Tone C	Tone B	Tone A
R10	Channel A Amplitude	/////	/////	/////	M	L3	L2	L1	L0
R11	Channel B Amplitude	/////	/////	/////	M	L3	L2	L1	L0
R12	Channel C Amplitude	/////	/////	/////	M	L3	L2	L1	L0
R13	Envelope Period	8-BIT Fine Tune E							
R14		8-BIT Coarse Tune E							
R15	Envelope Shape Cycle	/////	/////	/////	/////	CONT	ATT	ALT	HOLD
R16	I/O Port A Data Store	8-BIT PARALLEL I/O on Port A							
R17	I/O Port B Data Store	8-BIT PARALLEL I/O Port B							

Table III-11. PSG Register Array

4-2-3. Pin Function

DA7 - DA0 (input/output/high impedance): pins 30 -- 37 (AY-3-8910)
Data/Address 7 - 0:   pins 21 -- 28 (AY-3-8912)

These 8 lines comprise the 8-bit bidirectional bus used by the microprocessor to send both data and addresses to the PSG and to receive data from the PSG. In the data mode, DA7 -- DA0 correspond to Register Array bits B7 -- DA0 in conjunction with address inputs A9 and A8 form the high order address (chip select).

A8 (input): pin 25 (AY-3-8910)
          pin 17 (AY-3-8912)

A9 (input): pin 24 (AY-3-8910)
          (not provided on AY-3-8912)

Address 9, Address 8
These "extra" address bits are made available to enable the positioning of the PSG (assigning a 16 word memory space) in a total 1.024 word memory area rather than in a 256 word memory areas as defined by address bits DA7 - DA0 alone. If the memory size does not require the use of these extra address lines they may be left unconnected as each is provided with either an on-chip pull-down (A9) or pull-up (A8) resistor. In "noisy" environments, however it is recommended that A9 and A8 be tied to an external ground and +5V, respectively, if they are not to be used.

RESET (input): pin 23 (AY-3-8910)
              pin 16 (AY-3-8912)

For initialization/power-on purposes, applying a logic "0" (ground) to the Reset pin will reset all registers to "0". The Reset pin is provided with an on-chip pull-up resistor.

CLOCK (input): pin 22 (AY-3-8910)
pin 15 (AY-3-8912)

This TTL-compatible input supplies the timing reference for the Tone, Noise and Envelope Generators.

BDIR, BC2, BC1 (inputs): pins 27, 28, 29 (AY-3-8910)
pins 18, 19, 20 (AY-3-8912)

Bus DIRection, Bus Control 2, 1
These bus control signals are generated directly by GI's CP1600 series of microprocessors to control all external and internal bus operations in the PSG. When using a processor other than the CP1600, these signals can be provided either by comparable bus signals or by simulating the signals on I/O lines of the processor. The PSG decodes these signals as illustrated in the following:

BDIR	BC2	BC1	CP1600 FUNCTION	PSG FUNCTION
0	0	0	NACT	INACTIVE, See 010 (TAB) below.
0	0	1	ADAR	LATCH ADDRESS See 111 (INTAK) below
0	1	0	IAB	INACTIVE The PSG/CPU bus is inactive DA7-DA0 are in a high impedance state.
0	1	1	DTB	READ FROM PSG. This signal causes the contents of the register which is currently addressed to appear on the PSG/CPU bus. DA7-DA0 are in the input mode.
1	0	0	BAR	LATCH ADDRESS. See 111 (INTAK) below.
1	0	1	DW	INACTIVE. See 010 (IAB) above.
1	1	0	DWS	WRITE TO PSG. This signal indicates that the bus contains register data which should be latched into the currently addressed register. DA7--DA0 are in the input mode.
1	1	1	INTAK	LATCH ADDRESS. This signal indicates that the bus contains a register address which should be latched in the PSG. DA7 - DA0 are in the input mode.

While interfacing to a processor other than the CP1600 would simply require simulating the above decoding, the redundancies in the PSG functions vs. bus control signals can be used to advantage in that only four of the eight possible decoded bus functions are required by the PSG. This could simplify the programming of the bus control signals to the following, which would only require that the processor generate two bus control signals (BDIR and BC1, with BC2 tied to +5V) See fig. III-17.
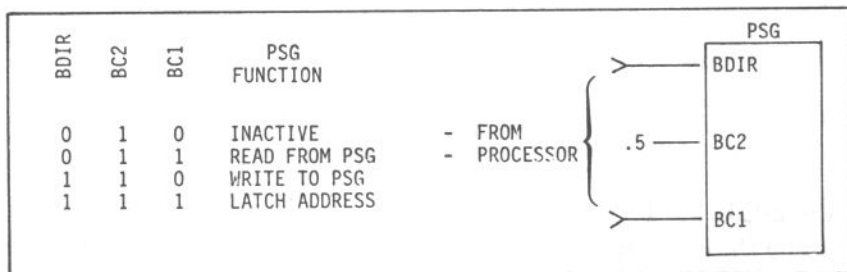

Fig. III-20.

ANALOG CHANNEL A, B, C (outputs): pins 4, 3, 38 (AY-3-8910)
                                                  pins 5, 4, 1 (AY-3-8912)

Each of these signals is the output of its corresponding D/A Converter, and provides an up to 1V peak-peak signal representing the complex sound waveshape dererated by the PSG.

IOA7 - IOA0 (input/output): pins 14 -- 21 (AY-3-8910)
                                         pins 7 -- 14 (AY-3-8912)

IOB7 --IOB0 (input/output): pins 6 -- 13 (AY-3-8910)
                                        (not provided on AY-3-8912)

Input/Output A7 -- A0, B7 -- B0

Each of these two parallel input/output ports provides 8 bits of parallel data to/from the PSG/CPU bus from/to any external devices connected to the IOA or IOB pins. Each pin is provided with an on-chip pull-up resistor, so that when in the 'input' mode, all pins will read normally high. Therefore, the recommended method for scanning external switches, for example, would be to ground the input bit.

TEST 1: pin 39 (AY-3-8910)
         pin 2 (AY-3-8912)

TEST 2: pin 26 (AY-3-8910)
         (not connected on AY-3-8912)

These pins are for GI test purposes only and should be left open - <u>do not</u> use as tie-points.

V_{cc}: pin 40 (AY-3-8910)
      pin 3 (AY-3-8912)

Nominal +5Volt power supply to the PSG.

V_{ss}: pin 1 (AY-3-8910)
    pin 6 (AY-3-8912)
Ground reference for the PSG.

4-3. Operation

Since all functions of the PSG are controlled by the host processor via a series of register loads, a detailed description of the PSG operation can best be accomplished by relating each PSG function to the control of its corresponding register. The function of creating or programming a specific sound or sound effect logically follows the control sequence listed:

Section	Operation	Register	Function
4-3-1.	Tone Generator Control	R0-R5	Program tone periods.
4-3-2.	Noise Generator Control	R6	Program noise period.
4-3-3.	Mixer Control	R7	Enable tone and/or noise on selected channels.
4-3-4.	Amplitude Control	R10-R12	Select "fixed" or "envelope-variable" amplitudes.
4-3-5.	Envelope Generator Control	R13--R15	Program envelope period and select envelope pattern.

4-3-1. Tone Generator Control

The frequency of each square wave generated by the three Tone Generators (one each for Channels A, B, and C) is obtained in the PSG by first counting down the input clock by 16, then by further counting down the result by the programmed 12-bit Tone Period value. Each 12-bit value is obtained in the PSG by combining the contents of the relative Coarse and Fine Tune registers, as illustrated in the following.

```
        Coarse Tune                              Fine Tune
         Register               Channel           Register

           R1                      A                RO
           R3                      B                R2
           R5                      C                R4

   ┌──┬──┬──┬──┬──┬──┬──┬──┐        ┌──┬──┬──┬──┬──┬──┬──┬──┐
   │B7│B6│B5│B4│B3│B2│B1│B0│        │B7│B6│B5│B4│B3│B2│B1│B0│
   └──┴──┴──┴──┴──┴──┴──┴──┘        └──┴──┴──┴──┴──┴──┴──┴──┘

      NOT
      USED

        ┌────┬────┬───┬───┬───┬───┬───┬───┬───┬───┬───┬───┐
        │TP11│TP10│TP9│TP8│TP7│TP6│TP5│TP4│TP3│TP2│TP1│TP0│
        └────┴────┴───┴───┴───┴───┴───┴───┴───┴───┴───┴───┘

        12-bit Tone Period (TP) to Tone Generator
```
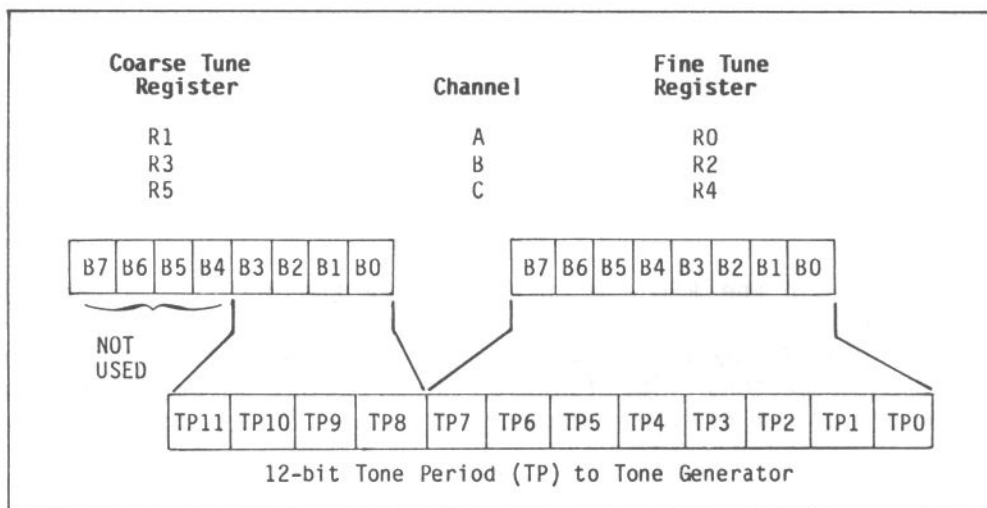
Fig. III-21.

Note that the 12-bit value programmed in the combined Coarse and Fine Tune registers is a period value - the higher the value in the registers, the lower the resultant tone frequency.

Note also that due to the design technique used in the Tone Period count-down, the lowest period, value is 000000000001 (divide by 1) and the highest period value is 111111111111 (divide by 4.095_{10})

The equations describing the relationship between the desired output tone frequency and the input clock frequency and Tone Period value are:

$$\text{(a) } f_T = \frac{f_{CLOCK}}{16TP_{10}} \qquad \text{(b) } TP_{10} = 256CT_{10} + FT_{10}$$

Where:
F_T = desired tone frequency

f_{CLOCK} = input clock frequency

TP_{10} = decimal equivalent of the Tone Period bits TP11--TP0

CT_{10} = decimal equivalent of the Coarse Tune register bits B3--B0 (TP11--TP8)

FT_{10} = decimal equivalent of the Fine Tune register bits B7--B0 (TP7--TP0)

From the above equations it can be seen that the tone frequency can range from a low of $\frac{f_{CLOCK}}{65.520}$ (wherein: $TP_{10} = 4.095_{10}$) to a high of $\frac{f_{CLOCK}}{16}$ (wherein: $TP_{10} = 1$). Using a 2MHz input clock, for example, would produce a range of tone frequencies from 30.5Hz to 125kHz.

To calculate the values for the contents of the Tone Period Coarse and Fine Tune registers, given the input clock and the desired output tone frequencies, we simply rearrange the above equations, yielding:

(a) $TP_{10} = \dfrac{f_{CLOCK}}{16 f_T}$ (b) $CT_{10} + \dfrac{FT_{10}}{256} = \dfrac{TP_{10}}{256}$

Example 1: $f_T = 1\text{KHz}$

$$f_{CLOCK} = 2\text{MHz}$$

$$TP_{10} = \frac{2 \times 10^6}{16(1 \times 10^3)} = 125$$

Substituting this result into equation (b):

$$CT_{10} + \frac{FT_{10}}{256} = \frac{125}{256}$$

$$\therefore \quad CT_{10} = 0 \qquad = 0000 \ (B3\text{--}B0)$$

$$FT_{10} = 125_{10} \qquad = 0111101 \ (B7\text{--}B0)$$

Example 2: $f_T = 100\text{ Hz}$

$$f_{CLOCK} = 2\text{MHz}$$

$$TP_{10} = \frac{2 \times 10^6}{16(1 \times 10^2)} = 1250$$

Substituting this result into equation (b):

$$CT_{10} + \frac{FT_{10}}{256} = \frac{1250}{256} = 4 + \frac{226}{256}$$

$$\therefore \quad CT_{10} = 4_{10} = 0100 \ (B3\text{--}B0)$$

$$FT_{10} = 226_{10} = 11100010 \ (B7\text{--}B0)$$

4-3-2. Noise Generator Control

The frequency of the noise source is obtained in the PSG by first counting down the input clock by 16, then by further counting down the result by the programmed 5-bit Noise Period value. This 5-bit value consists of the lower 5 bits (B4--B0) of register R6, as illustrated in the following.

```
┌─────────────────────────────────────────┐
│         Noise Period                     │
│         Register R6                      │
│                                          │
│  ┌──┬──┬──┬──┬──┬──┬──┬──┐               │
│  │B7│B6│B5│B4│B3│B2│B1│B0│               │
│  └──┴──┴──┴──┴──┴──┴──┴──┘               │
│   ╰──┬───╯ ╰─────┬──────╯                │
│     NOT      5-bit Noise Period (NP)     │
│     USED     to Noise Generator          │
└─────────────────────────────────────────┘
```
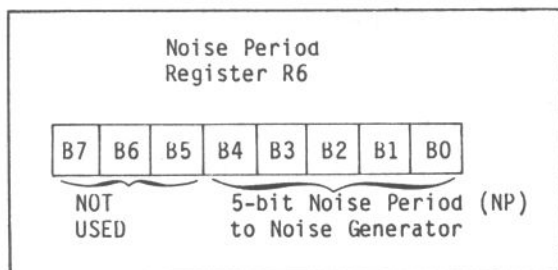
Fig. III-22.

Note that the 5-bit value in R11 is a period value - the higher the value in the register, the lower the resultant noise frequency. Note also that, as with the Tone Period, the lowest period value is 00001 (divide by 1): the highest period value is 11111 (divide by 31_{10}).

The noise frequency equation is:

$$f_N = \frac{f_{CLOCK}}{16 NP_{10}}$$

where: f_N = desired noise frequency

f_{CLOCK} = input clock frequency

NP_{10} = decimal equivalent of the Noise Period register bits B4--B0

From the above equation it can be seen that the noise frequency can range from a low of $\frac{f_{CLOCK}}{496}$ (wherein: $NP_{10} = 31_{10}$) to a high of $\frac{f_{CLOCK}}{16}$ (wherein: $NP_{10}=1$). Using a 2MHz input clock, for example, would produce a range of noise frequencies from 4kHz to 125kHz.

To calculate the value for the contents of the Noise Period register, given the input clock and the desired output noise frequencies, we simply rearrange the above equation, yielding:

$$NP_{10} = \frac{f_{CLOCK}}{16 f_N}$$

4-3-3. Mixer Control I/O Enable

Register 7, is a multi-function Enable register which controls the three Noise/Tone Mixers and the two general purpose I/O Ports.

The Mixers, as previously described, combine the noise and tone frequencies for each of the three channels. The determination of combining neither/either/both noise and tone frequencies on each channel is made by the state of bits B5--B0 of R7.

The direction (input or output) of the two general purpose I/O Ports (IOA and IOB) is determined by the state of bits B7 and B6 of R7.

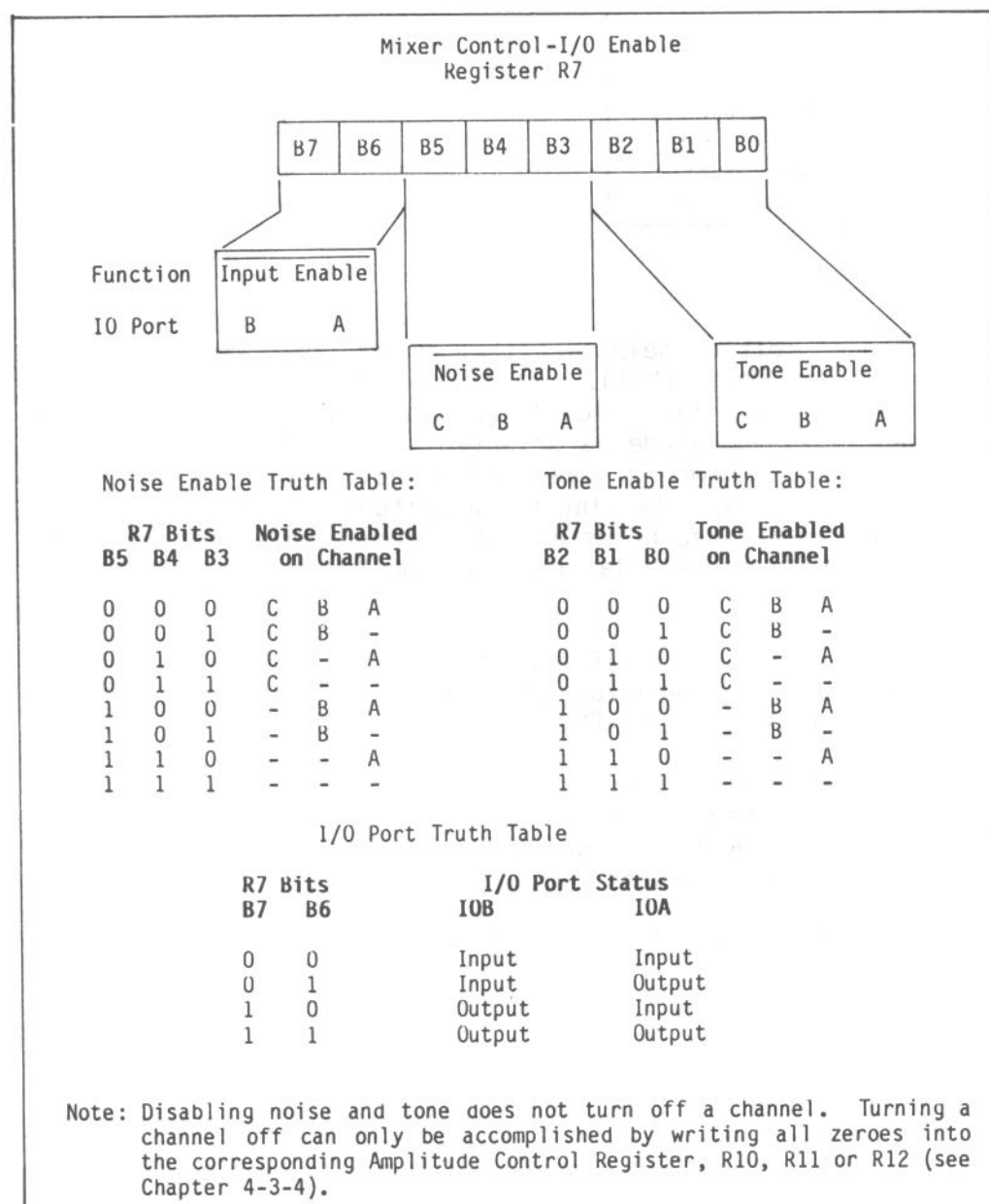These functions are illustrated in the following:

```
                    Mixer Control-I/O Enable
                         Register R7

            ┌────┬────┬────┬────┬────┬────┬────┬────┐
            │ B7 │ B6 │ B5 │ B4 │ B3 │ B2 │ B1 │ B0 │
            └────┴────┴────┴────┴────┴────┴────┴────┘

Function   ┌─────────────────┐
           │ Input Enable    │
IO Port    │   B        A    │
           └─────────────────┘
                       ┌──────────────────┐
                       │  Noise Enable     │
                       │  C    B    A      │
                       └──────────────────┘
                                        ┌──────────────────┐
                                        │  Tone Enable     │
                                        │  C    B    A     │
                                        └──────────────────┘
```

Noise Enable Truth Table:

R7 Bits			Noise Enabled on Channel		
B5	B4	B3			
0	0	0	C	B	A
0	0	1	C	B	-
0	1	0	C	-	A
0	1	1	C	-	-
1	0	0	-	B	A
1	0	1	-	B	-
1	1	0	-	-	A
1	1	1	-	-	-

Tone Enable Truth Table:

R7 Bits			Tone Enabled on Channel		
B2	B1	B0			
0	0	0	C	B	A
0	0	1	C	B	-
0	1	0	C	-	A
0	1	1	C	-	-
1	0	0	-	B	A
1	0	1	-	B	-
1	1	0	-	-	A
1	1	1	-	-	-

I/O Port Truth Table

R7 Bits		I/O Port Status	
B7	B6	IOB	IOA
0	0	Input	Input
0	1	Input	Output
1	0	Output	Input
1	1	Output	Output

Note: Disabling noise and tone does not turn off a channel. Turning a channel off can only be accomplished by writing all zeroes into the corresponding Amplitude Control Register, R10, R11 or R12 (see Chapter 4-3-4).

Fig. III-23.

4-3-4. Amplitude Control   (Registers R10, R11, R12)

The amplitudes of the signals generated by each of the three D/A
Converters (one each for Channel A, B and C) are determined by the
contents of the lower 5 bits (B4--B0) of registers R10, R11 and R12 as
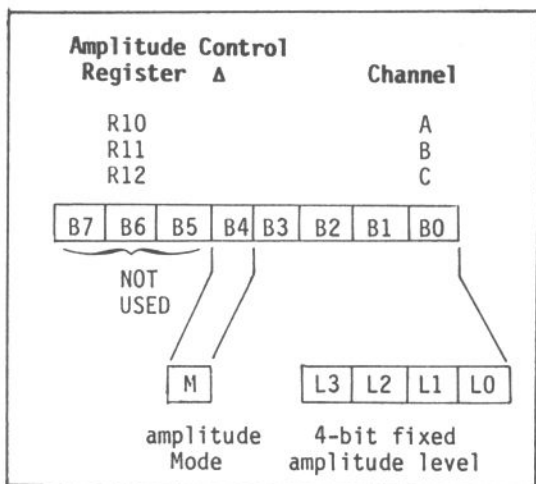illustrated in the following:

Fig. III-24.

The amplitude "mode" (bit M) selects either fixed level amplitude (M=0)
or variable level amplitude (M=1).   It follows then that bits L3--L0,
defining the value of a "fixed" level amplitude, are only active when
M=0.   When fixed level amplitude is selected, it is "fixed" only in the
sense that the amplitude level is under the direct control of the system
processor (via bits D3--D0).   Varying the amplitude when in this "fixed"
amplitude mode requires in each instance the direct intervention of the
system processor via an address latch/write data sequence to modify the
D3--D0 data.

When M=1 (select "variable" level amplitudes), the amplitude of each
channel is determined by the envelope pattern as defined by the Envelope
Generator's 4-bit output E3 E2 E1 E0.

The amplitude "mode" (bit M) can also be thought of as an "envelope
enable" bit, i.e. when M=0 the envelope is not used, and when M=1 the
envelope is enabled.   (A full description of the Envelope Generator
function follows in Section 4-3-5).

The full chart describing all combinations of the 5-bit Amplitude
Control is as follows:
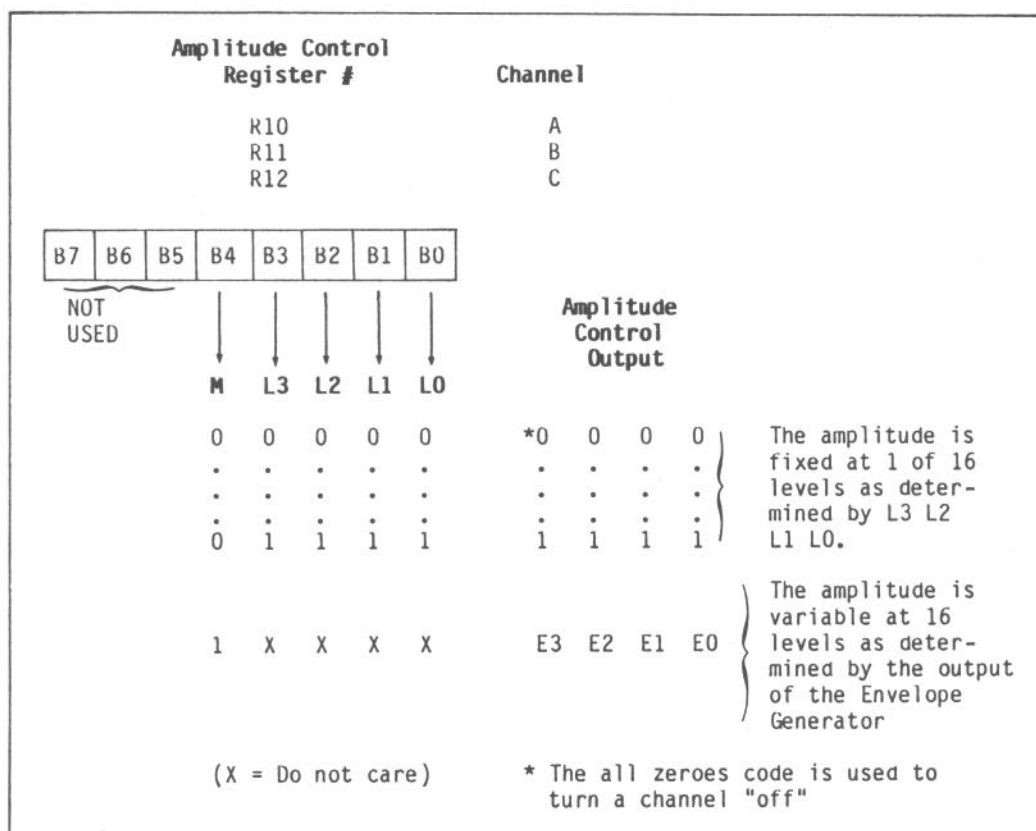
Fig. III-25.

Fig. III-26. graphically illustrates a selection of variable level (envelope-controlled) amplitude where the 16 levels directly reflect the output of the Envelope Generator. A fixed level amplitude would correspond to only one of the levels shown, with the level directly determined by the decimal equivalent of bits L3 L2 L1 L0.
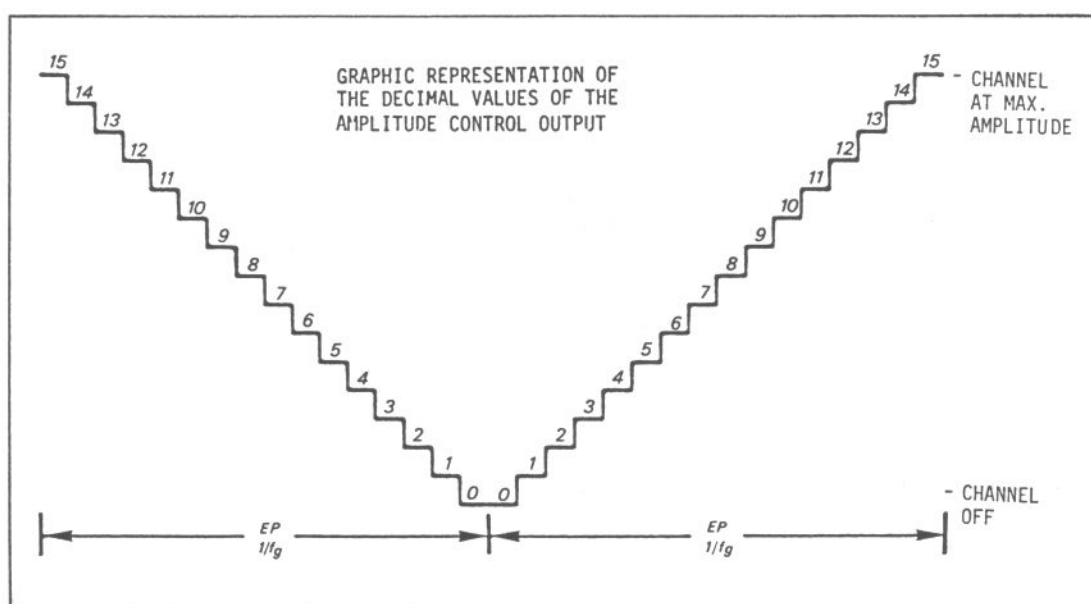

Fig. III-26. Variable Amplitude Control (M=1)

4-3-5. Envelope Generator Control (Registers R13, R14, R15)

To accomplish the generation of fairly complex envelope patterns, two
independent methods of control are provided in the PSG: first, it is
possible to vary the frequency of the envelope using registers R13 and
R14; and second, the relative shape and cycle pattern of the envelope
can be varied using register R15. The following paragraphs explain the
details of the envelope control functions, describing first the envelope
period control and then the envelope shape/cycle control.

4-3-5-1. Envelope Period Control (Registers R13, R14)

The frequency of the envelope is obtained in the PSG by first counting
down the input clock by 256, then by further counting down the result by
the programmed 16-bit Envelope Period value. This 16-bit value is
obtained in the PSG by combining the contents of the Envelope Coarse and
Fine Tune registers, all illustrated in the following:
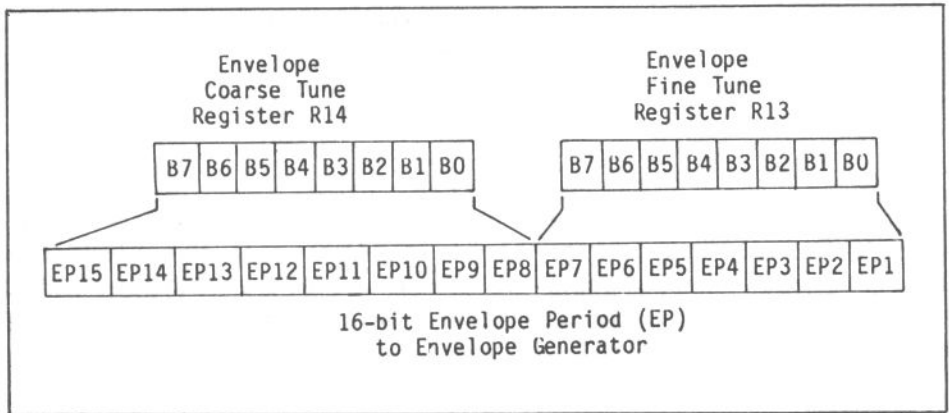
Fig. III-27.

Note that the 16-bit value programmed in the combined Coarse and Fine
Tune registers is a period value - the higher the value in the
registers, the lower the resultant envelope frequency.

Note also, that as with the Tone Period, the lowest period value is
000000000000001 (divide by 1); the highest period value is
1111111111111111 (divide by 65.535_{10}).

The envelope frequency equations are:

$$\text{(a)} \quad F_E = \frac{f_{CLOCK}}{256EP_{10}} \quad \text{(b)} \quad EP_{10} = 256CT_{10} + FT_{10}$$

Where: f_E = desired envelope frequency

f_{CLOCK} = input clock frequency

EP_{10} =decimal equivalent of the Envelope Period bits
EP15--EP0

CT_{10} =decimal equivalent of the Coarse Tune register
bits B7--B0 (EP15--EP8)

FT_{10} = decimal equivalent of the Fine Tune register bits
B7--B0 (EP7--EP0)

From the above equation it can be seen that the envelope frequency can range from a low of $\dfrac{f_{CLOCK}}{16.776.960_{10}}$ (wherein: $EP_{10} = 65.535_{10}$) to a high of $\dfrac{f_{CLOCK}}{256}$ (wherein: $EP_{10}=1$).

Using a 2MHz clock, for example, would produce a range of envelope frequencies from 0.12Hz to 7812.5Hz.

To calculate the values for the contents of the Envelope Period Coarse and Fine Tune registers, given the input clock and the desired envelope frequencies, we rearrange the above equations, yielding

$$(a)\ EP_{10} = \frac{f_{CLOCK}}{256f_E} \qquad (b)\ CT_{10} + \frac{FT_{10}}{256} = \frac{EP_{10}}{256}$$

Example: f_E = 0.5Hz

f_{CLOCK} = 2MHz

$$EP_{10} = \frac{2 \times 10^6}{256(0.5)} = 15.625$$

Substituting this result into equation (b):

$$CT_{10} + \frac{FT_{10}}{256} = \frac{15.625}{256} = 61 + \frac{9}{256}$$

$$CT_{10} = 61_{10} = 00111101\ (B7\text{--}B0)$$
$$FT_{10} = 9_{10} = 00001001\ (B7\text{--}B0)$$

4-3-5-2. Envelope Shape/cycle control (Register R15)

The Envelope Generator further counts down the envelope frequency by 16, producing a 16-state per cycle envelope pattern as defined by its 4-bit counter output, E3 E2 E1 E0. The particular shape and cycle pattern of any desired envelope is accomplished by controlling the count pattern (count up/count down) of the 4-bit counter and by defining a single-cycle or repeat-cycle pattern.

This envelope shape/cycle control is contained in the lower 4 bits (B3--B0) of register R15. Each of these 4 bits controls a function in the envelope generator, as illustrated in the following.


```
Envelope Shape/cycle
Control Register (R15)

┌──┬──┬──┬──┬──┬──┬──┬──┐   Function
│B7│B6│B5│B4│B3│B2│B1│B0│
└──┴──┴──┴──┴──┴──┴──┴──┘
 └────┬────┘              → Hold        ⎫
     NOT                  → Alternate   ⎬ To
     USED                 → Attack      ⎬ Envelope
                          → Continue    ⎭ Generator
```
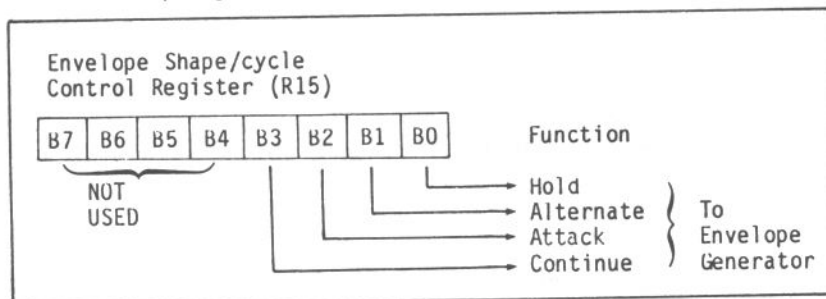
Fig. III-28. The definition of each function is as follows:

Hold   when set to logic "1", limits the envelope to one cycle, holding the last count of the envelope counter (E3--E0 = 0000 or 1111, depending on whether the envelope counter was in a count-down or count-up mode, repectively).

Alternate   when set to logic "1", the envelope counter reverses count direction (up-down) after each cycle.

NOTE: When both the Hold bit and the Alternate bit are ones, the envelope counter is reset to its initial count before holding.

Attack   when set to logic "1", the envelope counter will count up (attack) from E3 E2 E1 E0 = 0000 to E3 E2 E1 E0 = 1111; when set to logic "0", the envelope counter will count down (decay) from 1111 to 0000.

Continue   when set to logic "1", the cycle pattern will be as defined by the hold bit; when set to logic "0" the envelope generator will reset to 0000 after one cycle and hold at that count.

To further describe the above functions could be accomplished by numerous charts of the binary count sequence of E3 E2 E1 E0 for each combination of Hold, Alternate, Attack and Continue. However, since these outputs are used (when selected by the Amplitude Control registers) to amplitude modulate the output of the Mixers, a better understanding of their effect can be accomplished via a graphic representation of their value for each condition selected, as illustrated in Fig. III-29 and III-30.
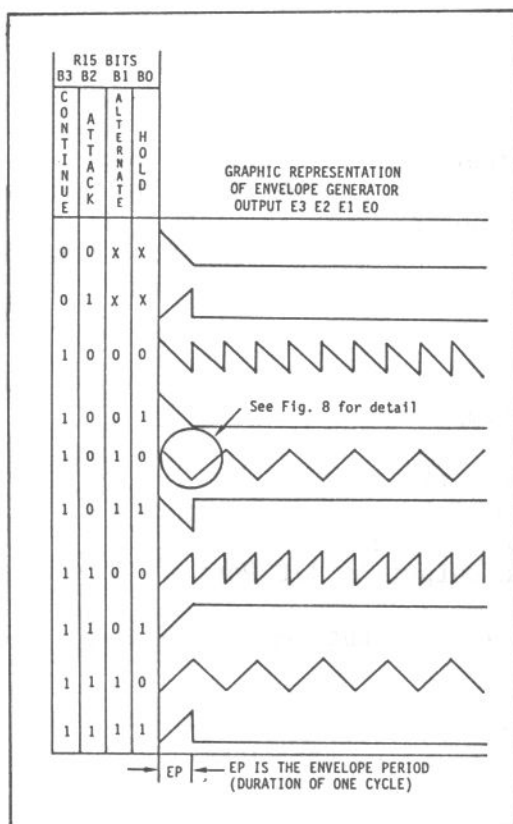
R15 BITS B3 (CONTINUE)	B2 (ATTACK)	B1 (ALTERNATE)	B0 (HOLD)	GRAPHIC REPRESENTATION OF ENVELOPE GENERATOR OUTPUT E3 E2 E1 E0
0	0	X	X	
0	1	X	X	
1	0	0	0	
1	0	0	1	See Fig. 8 for detail
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

EP — EP IS THE ENVELOPE PERIOD (DURATION OF ONE CYCLE)

Fig. III-29. Envelope Shape/Cycle Control

GRAPHIC REPRESENTATION OF THE DECIMAL VALUES OF THE ENVELOPE GENERATOR OUTPUT E3 E2 E1 E0

EP $1/f_g$ EP $1/f_g$
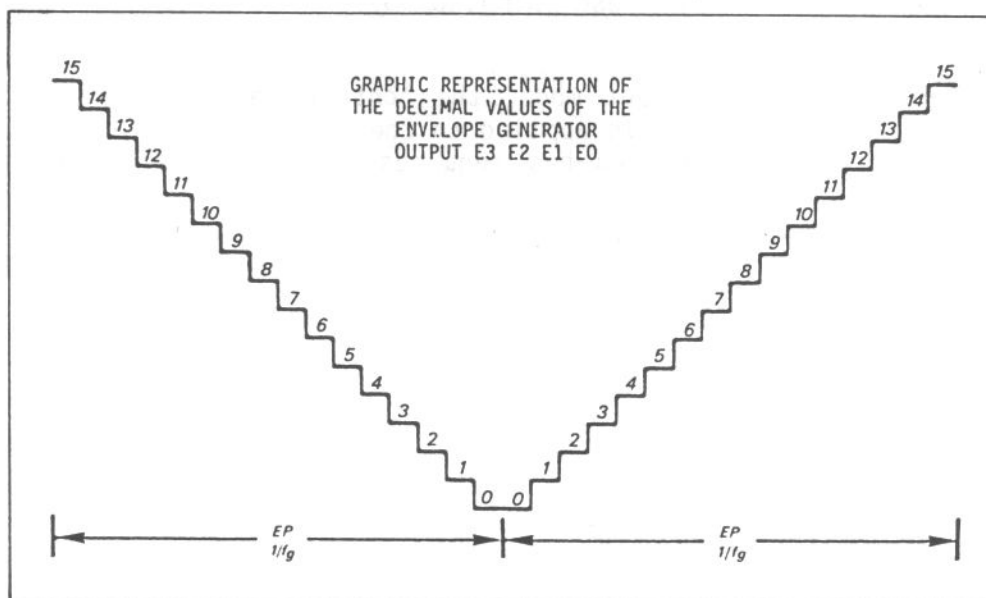
Fig. III-30. Detail of two Cycles of Fig. III-29.

4-3-6. I/O Port Data Store (Register R16, R17)

Registers R16 and R17 function as intermediate data storage registers
between the PSG/CPU data bus (DA0--DA7) and the two I/O ports
(IOA7--IOA0) and IOB7--IOB0). Both ports are available in the
AY-3-8910, only I/O Port A is available in the AY-3-8912. Using
registers R16 and R17 for the transfer of I/O data has no effect at all
on sound generation.

To output data from the CPU bus to a peripheral device connected to I/O
Port A would require only the following steps:

        1. Latch address R7 (select Enable register)
        2. Write data to PSG (setting B6 of R7 to "1")
        3. Latch address R16 (select IOA register)
        4. Write data to PSG (data to be output on I/O Port A)

To input data from I/O Port A to the CPU bus would require the
following:

        1. Latch address R7 (select Enable register)
        2. Write data to PSG (setting B6 to R7 to "0")
        3. Latch address R16 (select IOA register)
        4. Read data from PSG (data from I/O Port A)

Note that once loaded with data in the output mode, the data will remain
on the I/O port(s) until changed either by loading different data by
applying a reset (grounding the Reset pin), or by switching to the input
mode.

Note also that when in the input mode, the contents of registers R16
and/or R17 will follow the signals applied to the I/O port(s). However,
transfer of this data to the CPU bus requires a "read" operation as
decribed above.

PART IV

MSX-DOS

1. Introduction

The Microsoft Extended Disk Operating System (MSX-DOS) emulates the CP/M80 function calls (the 8-bit Digital Research's CP/M system) but contains the MS-DOS file formats (the 16-bit Microsoft's MS-DOS system). This part is intended to supply the technically oriented users with information about the structure, facilities and program interfaces of MSX-DOS.
It is assumed that the reader is familiar with the Z-80 architecture and instruction set.

This information will help programmers when running CP/M-80 software in a MSX-DOS environment.

2. The MSX-DOS Structure

The MSX-DOS consists of the following parts:

1. A Cold Boot record, residing on track Ø, sector 1 of every disk formatted by the FORMAT command. It is put on all disks in order to produce an error message if you try to start up the system with a non-MSX-DOS diskette in drive A:.

2. A ROM based BIOS interface module. This Basic Input Output System (BIOS) links the hardware environment to the standard MSX-DOS kernel.

3. The ROM based MSX-DOS kernel, providing a high-level interface for user programs.

4. The command processor (COMMAND. COM) which resides on the data area of the disk.

3. The MSX-DOS Disk Organization

1. The Sony 3.5" Floppy Disk (Single Sided, Double Density) is organized by the FORMAT command as follows:

   - 80 Tracks/side (0 - 79)
   - 9 sectors/track (1 - 9)
   - 512 bytes/sector
   The formatted storage capacity = 360 Kilobytes.

2. The cold boot loader resides on track 0, sector 1 as mentioned in the MSX-DOS Structure.

3. The 1st File Allocation Table (FAT #0) resides on track 0, sectors 2 and 3.
    A 2nd FAT (FAT #1) resides on track 0, sectors 4 and 5.
    There are two FAT's for redundancy reasons.
    They are normally identical except when they are being updated or if they have been corrupted in some fashion.
    Only one copy of these FAT's is kept in memory by MSX-DOS for each disk, and both copies on the disk are updated one after the other as required.

4. The remaining 4 sectors of track 0 (sectors 6, 7, 8 and 9) and the first 3 sectors of track 1 (sectors 1, 2 and 3) are used for the disk DIRECTORY.
    As each directory entry is 32 bytes in length, 112 directory entries can be made on a disk. The format of such an entry is as follows (byte offsets are in decimal).

    0 - 7   Filename (8 bytes)
            The first byte of this field indicates the status of this entry:

            Byte 0   Description
            00 hex   entry never used
            E5 hex   entry erased
            other   first character of the filename.

    8 - 10 File type (3 bytes)

    24 - 25 Date field (creation or last update)
            The month/day/year information is mapped in the bits as follows:

                    25   24
           ⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜⏜
           | Y | Y | Y | Y | Y | Y | Y | M | M | M | M | D | D | D | D | D |

           where D D D D D = day (1 - 31)
                 M M M M   = month (1 - 12)
                 Y Y Y Y Y Y = year (0 - 119)

           Note that the year number is an offset from year 1980

    26 - 27 Starting cluster: the relative cluster number of the first cluster in the file.
            Note that the first cluster for the data area is always cluster #002. This number is stored with the least significant byte first.

    28 - 31 file size in bytes. The first word (28 - 29) contains the low-order part of the size. Both words are stored with the least significant byte first.

- 90 -

5. The remaining sectors on track 2 (sector 4, 5, 6, 7, 8 and 9) and all the sectors of the other tracks are used for the DATA area.

4. MSX-DOS File Allocation

MSX-DOS uses "clusters" for storing data on a disk. Essentially, a cluster is a fixed-size block of disk sectors. Disk space is allocated in "clusters" and not in sectors.

For the SONY 3.5" F.D. following data are used:

- 354 clusters/disk in total.
- 351 clusters/disk for data storage.
- 2 sectors/cluster

As the sector size is 512 bytes, a cluster equals 1 KByte.
This implies that a program of 200 bytes will actually use 1 cluster or 1Kbyte of disk space.
A file of 1025 bytes (1KByte + 1) will need 2 clusters or 2 KByte of disk space.

The allocation of the clusters is stored in the FAT area of the disk. Each FAT entry (12 bits or 1.5 byte) corresponds to one "cluster" in the DATA area.

The FAT index in the DIRECTORY entry of a file indexes the first FAT entry for the file. This entry contains an index for the next FAT entry for the file and so on.

Any data record in a file can be found by computing the cluster index containing the data and searching the file's allocation chain in the FAT to find the appropriate cluster.

5. MSX-DOS Function Requests

The user requests a function by:

1. Placing a function number in the C register.
2. Supplying additional information in other registers for the specific function.
3. Executing a CALL 0005H instruction.

Unless otherwise specified, single-byte values are passed in E and double-byte values in DE. When MSX-DOS takes control, it switches to an internal stack. No registers are guaranteed to be preserved, MSX-DOS returns single-byte values in register A and double-byte values in register pair HL. In addition, the contents of register A will always equal those of register L, and the contents of B will equal H for those calls which have a CP/M-80 counterpart. A value of zero will be returned in A (as well as HL) for non-supported function numbers.

Function numbers are as follows. All values are in hex:

0 **Program Terminate**: All file buffers are flushed, but files which have been changed in length (but not closed) will not be recorded properly in the disk directory. Control transfers back to MSX-DOS. This is the same as JMP ØØØØH.

1 **Keyboard:** Waits for a character to be typed at the keyboard, then echoes the character to the screen and returns it in A. A console status check is done to check to Control-C, printer echo, and Control-S. These characters will not be passed through this call. Execution will be suspended until a character is typed, if none was waiting when the call was made. Carriage Return, Line Feed, Backspace, and Bell are echoed as such. Tabs are expanded.

2 **Screen Output**: The character in E is output to the screen in a fashion similar to Function 1. A console status check is performed after the character is output. A VT52 driver is built into the BIOS.

Microsoft MSX-DOS

Escape sequences are:

j	Clear screen
E	Clear screen
K	Erase to end of line
J	Erase to end of screen
l	(Lowercase L) Erase entire line
L	Insert a line
M	Delete a line
Y	Locate cursor
A	Up
B	Down
C	Right
D	Left
H	Home
x4	Block cursor
y4	Underscore cursor
x5	Cursor off
y5	Cursor on

3 **Auxiliary Input**: Waits for a character from the serial port and returns it in A. A console status check is done beforehand, suspending reading from this device if Control-S is detected.

4 **Auxiliary Output**: The character in E is sent to the serial port. A console status check is performed beforehand.

5 **Printer Output**: The character in E is output to the printer. As console status check is performed beforehand. Tabs are not expanded by the DOS.

6 **Direct Console IO**: If E is FFH, then A returns with keyboard input character if one is ready and zero flag is not set; otherwise, AZ is zero and zero flag is set. If E is not FFH, the E is assumed to have a valid character which is output to the screen. A console status check is not performed, passing through the values Control-C and Control-S.

7 **Direct Console Input**: Waits for a character to be typed at the keyboard, then returns the character in A. As with Function 6, no console status check is performed.

Note that CP/M-80 Function 7 is Set I/O Byte. I/O byte is not supported under MSX-DOS.

8 **Keyboard Input without Echo**: Identical to Function 1, without displaying the character.

Note that CP/M-80 Function 8 is Set I/O Byte. I/O byte is not supported under MSX-DOS.

9 **Display String**: On entry DE must point to a character string in memory terminated with a "$" (24H). Each character in the string will be displayed to the screen in the same form as Function 2, including console status check.

A **Buffered Keyboard Input**: On entry, DE points to an input buffer. The first byte must not be zero and specifies the number of characters the buffer can hold. Characters are read from the keyboard and placed in the buffer at the third byte. Reading the keyboard and filling the buffer continues until the carriage return is typed. If the buffer fills to the maximum, additional keyboard input is ignored and a bell is rung until carriage return is entered. The second byte of the buffer is set to the number of characters received excluding the carriage return (which is always the last character) unless the buffer was full prior to the carriage return. Editing of this buffer is described below and is somewhat different from CP/M-80.

--->	Copy one character
DEL	Skip one character
SEL	Copy until first occurrence of next character typed
CLS	Skip until first occurrence of next character typed
	Copy to end of line
,ESC	Kill new line
HOME	Re-edit line
<--, BS	Backspace one character
INS	Toggle insert mode

Note that CP/M-80 does not place the terminating carriage return in the buffer and that this byte will be uninitialized. However, because the count of characters does not include the carriage return, the count returned is the same as that under CP/M-80.

B **Keyboard Status Check**: If a character is available from the keyboard, A will return FFH. Otherwise A will be zero. If the available character is Control-C , Control-S, or printer echo, the appropriate action will be taken.

C **Disk Reset**: Flushes all file buffers. Files that have been changed in size and not closed will not be properly recorded in the disk directory until they are closed.

Unlike CP/M-80 this function need not be called before a disk change if all files which have been written have been closed.

E **Select Disk**: The drive specified in E (0 = A, 1 = B, etc.) is selected as the default drive.

Unlike CP/M-80, if the disk is changed the drive does not revert to Read Only status. Also, the number of drives is returned in A.

F **Open File**: On entry, DE points to an unopened file control block (FCB). The disk directory is searched for the named file and A returns FFH if it is not found. If it is found, A returns zero. The extent field is used and the record count field is appropriately filled in. The size of the file (in - bytes), the modification date and time are set in the FCB from information obtained from the directory. It is the application's responsibility to set the record size to the desired size if it uses the block read and write calls. It is also the application's responsibility to set the random record field and/or the extent and current record fields.

Unlike CP/M-80, there is no concept of partial or missing extents in MSX-DOS. It is not possible to create a file with "holes" in it. Therefore, this function may not fail in all the cases CP/M-80 would. Note that different directory information is copied into bytes 10-1F4 of the FCB than in CP/M-80. Also, MSX-DOS always returns a zero if successful, whereas CP/M-80 returns a number from 0 to 3, indicating the position within the logical directory sector of this file.

10 Close File: This function must be called after file writes to ensure all directory information and the File Allocation Table is updated. On entry, DE points to an opened FCB.

Unlike CP/M-80, MSX-DOS assumes that the disk has been changed and A returns FFH. Otherwise, the directory is updated to reflect the status in the FCB and A returns zero.

11 Search First: On entry, DE points to an unopened FCB. The disk directory is searched for the first matching name (the name could have "?"s indicating any letter matched) and if none is found, A returns FFH. Otherwise 33 (as opposed to CP/M-80's 128) locations at the disk transfer address contain a valid unopened FCB with the first byte indicating the drive number used (1=A, etc.) and A returns a zero. The extent field returned will match the one that was searched for, with the record count initialized appropriately.

Unlike CP/M-80, the directory position returned is always zero (with CP/M-80 it can be 0 to 3). Also, if the drive field contains a "?", the file name and extent field are ignored, causing anything to match. Deleted directory entries are not accessible and looking beyond the first directory entry at the disk transfer address will not work. If the extent field is a "?", the resultant extent field is set to the highest extent and record count.

12   **Search Next:**   After a Function 11 has been called and has found a match, Function 12 may be called to find the next match to an ambiguous request ("?"s in the search filename).   Both inputs and outputs are the same as Function 11.   No intervening calls should be made between Search First and Search Next or successive Search Next requests.   Search Next assumes the address of the FCB used in Search First.

13   **Delete File:** On entry, DE points to an unopened FCB.   All matching directory entries are deleted.   If no directory entries match, A returns FFH.   Otherwise, A returns zero.

14   **Sequential Read:** On entry, DE points to an opened FCB.   The record addressed by the current extent and current record is loaded at the disk transfer address, then the record number is incremented.   If end-of-file is encountered,   A returns 01H.   A returns zero if the transfer was completed successfully.

Unlike CP/M-80, which does not recognize record sizes other than 80H, partial records are zero-filled.

15   **Sequential Write:** On entry, DE points to an opened FCB.   The record addressed by the current extent and current record is written from the disk transfer address, then the record number is incremented. If the disk is full, A returns with 01H.   A returns zero if the transfer was completed successfully.

Note that in the case of records smaller than sector sizes, the sector is buffered up for an eventual write when a sector's worth of data is accumulated.

16   **Create File:** On entry, DE points to an unopened FCB.   The disk directory is searched for an empty directory entry, and A returns FFH if none is found.   Otherwise the entry is initialized to a zero length file, the file is opened (see Function F), and A returns zero.

Unlike CP/M-80, MSX-DOS always returns zero as the directory code, instead of 0 to 3.   If the file already exists and the extent field contains a zero, MSX-DOS deletes the existing file.   If the extent field is non-zero, the DOS opens the file and points to that extent.

17   **Rename File:** On entry, DE points to a modified FCB which has a drive code and filename in the usual position, and a second filename starting 6 bytes after the first (DE+11H) in what is normally a reserved area.   Every matching occurrence of the first is changed to the second (with the restriction that two files cannot have the same name and extension).   If no match was found,   A returns FFH.

Wild cards may be used in both source and destination files. If "?" appears in the second name, then the corresponding position in the original name will be unchanged.

18 Login Vector: Returns 1 bit for all drives on the system in HL, with drive A the low-order bit.

19 Current Disk: A returns with the code of the default drive (0=A, etc.).

1A Set Disk Transfer Address: The disk transfer address is set to DE. On a Disk Reset (Function D) the disk transfer address is reset to 80H.

1B FAT Address: A drive code is passed in E. On return, A contains the number of sectors per cluster for the default drive or FFH if the drive number is invalid. BC is sector size. DE is the number of clusters on the disk. HL is the number of free clusters. IY points to the FAT. IX points to the BPB.

CP/M-80 returns with HL pointing to a bit vector of free allocated clusters. The cluster size can be obtained from CP/M-80 Function 1F. No such structure exists within MSX-DOS.

1C Write Protect Drive: NOT IMPLEMENTED. This call will return with no processing.

1D Get R/O Vector: NOT IMPLEMENTED. This call with return with zeros, indicating no drives write-protected.

1E Set File Attributes: NOT IMPLEMENTED.

1F Get Disk Parameter Address: NOT IMPLEMENTED.

20 Set/Get User Code: NOT IMPLEMENTED

21 Random Read: On Entry, DE points to an opened FCB. The current extent and current record are set to agree with the random record field. This record is loaded at the disk transfer address. If end-of-file is encountered, A returns 01H. Otherwise, A returns zero if successful.

Unlike CP/M-80, which does not support them, partial records are zero-filled. The high-byte (r2) need not be zero and is always used, which addresses files up to 2 gigabytes.

22 Random Write: On entry, DE points to an opened FCB. The current extent and current record are set to agree with the random record field. This record is written from the disk transfer address. A returns 01H if the disk is full. Otherwise, A returns zero.

As with Sequential Write, the sector is buffered if the record size is smaller than the sector size.

23 File Size: On entry, DE points to an unopened FCB. The disk directory is searched for the first matching entry and if none is found, A returns FFH. Otherwise, the random record field is set with the size of the file in records, and A returns zero.

Unlike CP/M-80, MSX-DOS supports ambiguous filenames, using the first match.

24 Set Random Record: On entry, DE points to an opened FCB. This function sets the random record field to the same file address as the current extent and current record fields.

25 Disk Reset: NOT IMPLEMENTED. This call will return after no processing.

26 Random Block Write: Essentially the same as Function 27 above, except for writing and a write protect indication. If there is insufficient space on the disk, A returns 01H and no records are written. If HL is zero upon entry, no records are written, but the file is set to the length specified by the random record field, whether longer or shorter than the current file size. (Allocation units are released or allocated as appropriate.)

27 Random Block Read: On entry, DE points to an opened FCB, and HL contains a record count which must not be zero. The specified number of records (in terms of the record size field) are read from the file address specified by the random record field (3 bytes if sector size is greater than or equal to 40H; otherwise 4 bytes) into the disk transfer address. If end-of-file is reached, A returns 01H, zero filling any partial record. A returns zero on a successful read. In any case, HL returns the actual number of record read, and the random record field is set to address the next record.

28 Zero Fill Random Write: Same as call 22 except whenever a write request would extend a file contiguously, the space in between is zeroed as well as allocated.

29 NOT IMPLEMENTED

2A Get Date: Returns the date in DE. HL has the year, D has the month (1=Jan, etc.) and E has the day. A has the day of the week (0=Sun). If the time clock changes to the next day, the date will be adjusted accordingly, taking into account the number of days in each month and leap years.

2B Set Date: On entry, DE and HL must contain a valid date as described in Function 2A above. If the date is valid, A returns zero; otherwise, A returns FFH.

2C Get Time: Returns time of day. H has the hours, L has the minutes, D has the seconds, and E has the hundredths of seconds. This form is easily converted to a printable form, yet it can also be used to calculate (e.g. subtracting two times).

2D Set Time: On entry, DE and HL must contain a valid time as described in Function 2C. If the time is valid, A returns zero; otherwise A returns FFH.

2E Verify Read After Write: Supported

2F Direct Sector Read: Supported

30 Direct Sector Write: Supported

6. MSX-DOS FCB Format

The following explains the MSX-DOS File Control Block Format. The FCB is a 37-byte data area, with the following structure.

Offset	Function
0	Drive number. 0=default, 1=A, 2=B, ...
1-8	Filename left justified with trailing blanks. All 8 bits of characters are significant. If the name of a device (PRN, CON) is placed here, do not include the optional colon.
9-B	Filename extension, left justified with trailing blanks (can be all blanks). All 8 bits of characters are significant.
C	Extent field. Used on Open, Create, Search, Sequential Read, and Sequential Write. Set by Random Read and Random Write.
D	S1. Reserved.
E	S2. An 8-bit extension of the extent field. Zeroed on Open, Create, and Search First. Used as low byte of record size for Block Read and Block Write.
F	Record Count. Normally 8DH, but set to number of 128-byte records left in extent after Open and Create. Incremented as appropriate by Sequential Write and Random Write. Used as high byte of record size for Block Read and Block Write.
10-13	File size in bytes. In this 2-word field, the first word is the low order part of the size.
14-15	Date the filenames were created or last modified. Bits F-9 (year) = 0-119 (1980-2099) Bits 8-5 (month) = 1-12 Bits 4-0 (day) = 1-31

16-17   Time the file was created or last modified.

Bits F-B (hours) = 0-23
Bits A-5 (minutes) = 0-59
Bits 6-0 (seconds) = 0-29 (two second increments)

18-1F   Reserved

20   Current relative record number (0-127) within the current block. You must set this field before doing sequential read/write operations to the disk. This field is not initialized by the Open function call.

21-24   Relative record number relative to the beginning of the file, starting with zero. You must set this field before doing random read/write operations to the disk. This field is not initialized by the Open function call.

If the record size is less than 64 bytes, both words are used. Otherwise, only the first 3 bytes are used. Note that if you use the File Control Block at 5CH in the program segment, the last byte of the FCB overlaps the first byte of the unformatted parameter block.